NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

TRI-LEVEL OPTIMIZATION MODELS TO DEFEND CRITICAL INFRASTRUCTURE

by

Pablo Alvarez San Martin

September 2007

Thesis Advisor: Kevin Wood
Second Reader: Javier Salmeron

Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.			
1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** September 2007	**3. REPORT TYPE AND DATES COVERED** Master's Thesis	
4. TITLE AND SUBTITLE Tri-Level Optimization Models to Defend Critical Infrastructure		**5. FUNDING NUMBERS**	
6. AUTHOR(S) Pablo Alvarez San Martin			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000		**8. PERFORMING ORGANIZATION REPORT NUMBER**	
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A		**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**	
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited		**12b. DISTRIBUTION CODE**	

13. ABSTRACT (maximum 200 words)

This thesis develops and solves a tri-level optimization model to plan the optimal defense of an infrastructure from intelligent attack. We assume that a "defender" will first use limited defensive resources to protect system's components; then, an intelligent adversary ("attacker") will use limited offensive resources to attack unprotected components in order to inflict maximum damage to the system. The defender guides system operation with an optimization model, so increased operating cost equates to damage. This leads to a tri-level "defender-attacker-defender" model (DAD), where the second "defender" means "defender as system operator."

The general DAD is NP-hard and requires decomposition to solve. We develop four decomposition algorithms: direct, nested, reformulation-based, and reordering-based. The reordering-based algorithm computes an optimistic bound by reordering the stages of the DAD, and the reformulation-based algorithm uses subproblems that resemble standard capacity-interdiction models. Computational tests on generic instances of "defending the shortest path" (DSP) show the nested and reformulation-based algorithms to be twice faster than the first, on average.

A hypothetical instance of DSP provides a concrete illustration: A Spanish marine unit, in an emergency deployment, must defend its base-to-port route against potential terrorist attacks.

14. SUBJECT TERMS Critical infrastructure, tri-level optimization models, mixed-integer linear program, homeland security			**15. NUMBER OF PAGES** 103
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UU

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

TRI-LEVEL OPTIMIZATION MODELS TO DEFEND CRITICAL INFRASTRUCTURE

Pablo Alvarez San Martin
Lieutenant Commander, Navy
Spanish Naval Academy, 1993

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
September 2007

Author: Pablo Alvarez San Martin

Approved by: Kevin Wood
Thesis Advisor

Javier Salmeron
Second Reader

James N. Eagle
Chairman, Department of Operations Research

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

This thesis develops and solves a tri-level optimization model to plan the optimal defense of an infrastructure from intelligent attack. We assume that a "defender" will first use limited defensive resources to protect system's components; then, an intelligent adversary ("attacker") will use limited offensive resources to attack unprotected components in order to inflict maximum damage to the system. The defender guides system operation with an optimization model, so increased operating cost equates to damage. This leads to a tri-level "defender-attacker-defender" model (*DAD*), where the second "defender" means "defender as system operator."

The general *DAD* is NP-hard and requires decomposition to solve. We develop four decomposition algorithms: direct, nested, reformulation-based, and reordering-based. The reordering-based algorithm computes an optimistic bound by reordering the stages of the *DAD*, and the reformulation-based algorithm uses subproblems that resemble standard capacity-interdiction models. Computational tests on generic instances of "defending the shortest path" (DSP) show the nested and reformulation-based algorithms to be twice faster than the first, on average.

A hypothetical instance of DSP provides a concrete illustration: A Spanish marine unit, in an emergency deployment, must defend its base-to-port route against potential terrorist attacks.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

I thank my thesis advisor, Professor Kevin Wood, for his expert guidance, countless hours of dedication, and timely revisions. Without his support, this thesis could not have been conceived, nourished, or completed.

I, also, thank my second reader, Professor Javier Salmeron, for his critical view, and thorough revision. His valuable comments improved the quality of this thesis.

Professor Matthew Carlyle ("network flows" class) and Cap. Matthew Desmon (USMC)/Lt. Casey Mahon (USN) (our class project) enlightened me with ideas that helped with approaching this thesis with a clearer perspective.

Finally, I thank my wife, Maria, and my daughter, Paula, for their understanding and constant support during these past months. They did their best to prevent me from laptop addiction.

THIS PAGE INTENTIONALLY LEFT BLANK

EXECUTIVE SUMMARY

This thesis develops and solves tri-level optimization models to solve the problem of defending a generic system from intelligent attack. We assume that a "defender" will use limited resources to protect system components, and an intelligent adversary ("attacker") will use limited offensive resources to inflict the maximum possible damage on the defended system. The tri-level model becomes a "defender-attacker-defender" model (*DAD*). The defender guides his system operation with an optimization model, so damage is measured in terms of increased operations cost.

We develop a general *DAD* model that is, apparently, solvable only through decomposition. Thus, we devise a decomposition Algorithm 1 for *DAD*. The master problem for this decomposition resembles a master problem for a standard Benders decomposition, but the subproblem, instead of being a standard linear program (LP), is a mixed-integer linear program (MIP). Algorithm 1 solves this subproblem directly, i.e., with LP-based branch-and-bound, while Algorithm 1A solves it by Benders decomposition. Thus, Algorithm 1A may be viewed as a nested decomposition algorithm.

Algorithm 1A proves to be almost twice as fast as Algorithm 1 (1.87 times faster, on average) on test problems that involve "defending the shortest path" (DSP). DSP represents a situation in which the defender-operator needs to solve a minimization problem to operate the system optimally (e.g., transiting from station A to station B in the quickest time); the attacker seeks to maximize the defender's shortest path by interdicting some components of the system (e.g., cutting down some segments that lie in the A-B path); and the defender has to minimize the largest traversal distance after the attacker's best attack.

A *capacity-interdiction* formulation provides an alternative modeling approach to the same *DAD* model. This formulation is somewhat more complex than the formulation of the standard decomposition for *DAD*, and is also more difficult to implement. However, it proves to be almost as fast as nested decomposition (Algorithm 1A) for the test cases based on DSP.

Interchanging the order of the first two levels of optimization in the model, that is, converting "min-max-min" into "min-min-max," can provide an optimistic lower bound on the optimal objective-function value. We give the advantage to the defender who gets to see the attacker's plan before defending the system and operate it. The quality of the bound provided by *ADD* may be poor, but it can be improved by giving extra resource to the attacker, creating "ADD^{+}."

When the defender-operator solves a shortest-path problem, and the restrictions on the resources are given by simple knapsack constraints, *ADD* (and ADD^{+}) can be solved as a special network-interdiction problem called "maximizing the shortest path" (*MXSP*). In *MXSP,* the original system is expanded in levels, as many as the number of units of defensive resource, and each jump between levels corresponds to a defensive action taken by the defender.

A practical example based on DSP illustrates the effectiveness of the models and algorithms developed in this thesis. A small, Spanish infantry unit must traverse from the Marine Corps' headquarters to a nearby naval base for immediate deployment, using the road network in Cadiz Bay, Spain. A terrorist group can interdict up to six road segments, and the Marine Command can defend against these attacks by protecting 10 segments, by means of armed patrols. The road network built to represent this example contains approximately 200 nodes and 630 arcs. The solution provides the armed patrols optimal allotment and the time that would take the infantry unit to get to the naval base in the worst case scenario.

I. INTRODUCTION

This thesis addresses the problem of defending a critical infrastructure (or system) from intelligent attack by developing and applying tri-level optimization models. We assume the system's operator, called "defender," will use limited resources to protect (defend) system components. Subsequently, an intelligent adversary will use limited offensive resources to inflict maximum damage to the defended system. The defender guides system operation with a minimizing optimization model, so increased operating costs equate to damage. This leads to a tri-level "defender-attacker-defender" model (*DAD*), where the second "defender" means "defender as system operator." We explore, propose and implement different tri-level models to solve *DAD*.

This chapter defines and gives examples of critical infrastructure. It specifies a model of behavior for the defender, attacker, and their interactions, and it introduces a general framework under which these tri-level models will be developed.

A. CRITICAL INFRASTRUCTURE

Especially after 2001, governments have devoted much time and effort to identify critical infrastructure (CI) and to assess the impact on their nation's wealth that disruptions to that CI might have. The USA Patriot Act (U.S. Senate and House of Representatives 2001) defines CIs as:

> … systems and assets, whether physical or virtual, so vital to the United States that the incapacity or destruction of such systems and assets would have a debilitating impact on security, national economic security, national public health or safety, or any combination of those matters.

The National Strategy document for Homeland Security (Department of Homeland Security 2002) identifies thirteen sectors in which most of CI systems may be framed, including agriculture, food, water, public health, energy, transportation, information, and telecommunications.

The Department of Homeland Security (DHS) is responsible for keeping the national infrastructure inventory updated with basic information about the systems, including: the elements involved, either human, physical or cyber; services provided;

dependencies; and interdependencies. (The U.S. Office of the Inspector General stated that, as of January 2006, the National Infrastructure Database already contained up to 77,069 assets ranging from gas stations and retail outlets to nuclear plants and water distribution systems; see Inspector General 2006.)

Listing the systems is only the first step. A comprehensive vulnerability analysis is important to enable authorities to evaluate the effects of potential attacks and to invest to protect or harden system components. This is being accomplished by setting a common methodology for risk assessment, the Risk Management Framework, provided by the National Infrastructure Protection Plan ("NIPP"; see Department of Homeland Security 2002).

Many of these systems were built to be cost-effective, which implies they often provide only a minimum level of redundancy to satisfy demand or other requirements. They may reasonably well handle disruptions caused by random degradation of physical components, accidents due to mechanical or human failure, and acts of nature. However, these CIs may not show robustness against an intelligent attack that destroys critical components. For example, three bridges crossing a river in a populated city may handle the traffic between both sides even when one of them is closed due to a major accident. However when all three are intentionally put out of service, the crossing traffic ceases completely. Traditional vulnerability analysis needs the perspective of the terrorist threat to capture the behavior and response of a particular system under a new set of circumstances.

The standardized method for vulnerability assessment (Department of Homeland Security 2006) starts by analyzing a CI in terms of numerical measures of *threat* (t), *consequence* (c), and *vulnerability* (v) for individual system components. *Threat* reflects the likelihood that a component will suffer a terrorist attack. *Consequence* reflects the reduction in the system's performance given a successful attack on a component (the greater the reduction in performance, the greater is the consequence). *Vulnerability* reflects the likelihood that, if an attack occurs, the component will be partially damaged, incapacitated, or destroyed.

Every component i of a particular CI system is evaluated and given numerical values t_i, c_i and v_i, representing threat, consequence and vulnerability, respectively. Then, a measure of *risk (r)* is estimated for a particular system. In general $r = f(t,c,v)$, but typically, $r = t \times c \times v$. Components are then prioritized with respect to the estimated risk and this ordered list eventually provides the decision-maker with a picture to help him decide which components should be given priority for protection, where to allocate resources, what protective programs should be instituted, and what the appropriate level of investment in programs should be.

The protective programs may seek to prevent any potential attack by taking specific actions on the elements of the system subject to protection (e.g., building stronger fences along the perimeter of key installations, increasing the number of armored patrols on any given border segment), or to reduce its effects (e.g., by training emergency-response teams). However, this thesis primarily focuses on physically protecting CI components from attack.

The analysis of those protective actions and their impact on the risk mitigation, eventually guides the investments used in a particular program. Sometimes no analysis is available, and the program manager implements defensive measures by just following the guidelines of a written manual (FEMA 2007).

How should a limited defensive budget be spent to protect a single CI system as well as possible? The current methods utilized to guide investments into protective programs are based on priorities on individual components, but ignores component interactions. We propose the modeling and solution of an optimal defensive plan obtained from the solution of a tri-level optimization model.

We are interested in many different systems, but these are often modeled as networks, where some commodity (e.g., electricity, water) must be moved from one or more points to one or more other points, while following the topology of the underlying network and laws of physics. For example, consider the U.S. Strategic Petroleum Reserve (Department of Energy 2007): This system can be represented as a network in which

storage sites represent source nodes; pumping stations represent transit nodes; refineries and shipping ports represent sink nodes; and pipelines correspond to arcs that connect the various nodes.

We assume that the operator of the infrastructure, (henceforth called the *defender-operator*), operates his system following guidance from an optimization model ("Defender Model," or "*D*"), specifically by solving a linear program (LP). For example, the SPR management office, as the defender-operator, operates the system during an oil emergency, and possibly after an attack on the system, to ensure that enough oil flows from storage sites to meet demands.

Before defending the system, the protecting agency (henceforth called *defender*) needs to know how the terrorist organization (henceforth called the *attacker*) is going to attack it. We assume that the attacker, according to his resources, seeks to maximally disrupt a system's operations by interdicting certain components. Thus, the attacker solves a bi-level optimization model, Attacker-Defender (*AD*), where "*D*" denotes the defender-operator's optimization model mentioned above, solved at the inner level (See Brown, Carlyle, Salmeron and Wood 2005). There is no uncertainty about the attacker's resources, or what the effect of attacks might be. This is important because deterministic interdiction problems, as presented in this paper, rely on accurate information. Otherwise, we would be dealing with a stochastic interdiction problem (Cormican, Morton and Wood 1998), which is beyond the scope of this thesis.

As opposed to Cournot models (Cournot 1838), where both opponents move simultaneously, we model the opponents' interactions following the rules of a Stackelberg game (Stackelberg 1952). The *leader* (attacker) plays first by interdicting the system in an optimal way, and then, the *follower* (defender) observes the actions taken by the leader and makes his best choice. In most of the economic models played by the Stackelberg rules, any non-optimal solution adopted by the follower that deviates from equilibrium may hurt not only himself, but also, the leader. However, this is not the case here: If the attacker makes a non-optimal move, the results cannot cause worse damage

than foreseen by the solution of the bi-level model. Also, the defender-operator has the option to improve his cost by choosing another course of action which takes advantage of the attacker's neglect.

One of the key assumptions of these games is *perfect information*. Decisions taken by one player are based on complete knowledge of the actions that will be taken by the other. Furthermore, we assume that the attacker has perfect information about the system. This means that no components of the system are hidden to the attacker and, essentially, both players deal with the same problem. This leads to sensibly conservative damage assessments for the defender: The attacker can cause no more disruption to the system than the worst case identified by the solution of *AD*.

The solution of *AD* provides valuable information to the defender-operator. First, it points out critical components of the system. Second, it lays the groundwork for the next embellishment of the Stackelberg game, the addition of a level of active defense: Using a tri-level model, the defender seeks to minimize the maximum damage an attacker can inflict to the system when it is operated optimally.

The defender therefore, needs to solve a Defender-Attacker-Defender model *DAD* (See Brown, Carlyle, Salmeron and Wood 2005). Observe that, in order to differentiate the two roles that the defender plays in this model, a distinction has been made between defender and defender-operator. The former defends the system, while the latter operates it optimally.

Brown, Carlyle, Salmeron and Wood (2005) describe new bi-level models to solve the problem of defending CI, applying these models to electrical power grids, subways, airports and other systems. These authors also introduce the idea of embedding a given *AD* model in a tri-level *DAD* and state that this type of problem solves only with "extreme difficulty."

Brown, Carlyle, Salmeron and Wood (2006) formulate and solve an electrical-grid protection problem with a tri-level *DAD* model. However, as opposed to *AD* problems, full-scale tri-level problems cannot be solved yet.

This thesis proposes a general framework for *DAD* models represented as *tri-level mixed-integer linear programs* (TLMIPs), proposes several solution methods for such models, and investigates the computational behavior of these methods. Medium-size problems are implemented and solved by these models.

In general, the inner optimization problem of any TLMIP developed in this thesis is an LP for the defender's system-operation model, and resource constraints on system defense and attack will be fairly simple, such as knapsack constraints.

B. THESIS OUTLINE

Chapter II proposes and develops a general tri-level *DAD* model, and a *capacity-interdiction DAD,* which reformulates the basic *DAD* probing other solving methodologies.

Chapter III formulates an *ADD* model (Attacker-Defender-Defender model), where the two outer optimization layers, i.e., "DA," have been interchanged for the ultimate purpose of bounding, and eventually solving, *DAD*. Interchanging the order of these two levels gives the advantage to the defender, so *ADD* yields a lower bound for a min-max-min *DAD*. A method is also described to add resource for the attacker in order to tighten the bound. Provided this bound is tight enough and easy to obtain, *ADD* can be incorporated in the *DAD* decomposition method to accelerate the convergence of the algorithm.

Chapter IV presents different solution algorithms for these models and the implementation for the "Defending the Shortest-Path" problem (DSP).

Chapter V presents computational results from testing hypothetical network examples of different size and shape against the aforementioned algorithms.

Finally, Chapter VI illustrates a deployment protection problem of a Spanish Marine Corp Special Operation Forces (SOF) unit. This exercise requires the solution of DSP, using a tri-level DAD model, for a small Infantry entity who must traverse from its home base in San Fernando to the Naval Base in Rota for emergency deployment.

II. FORMULATIONS FOR THE TRI-LEVEL "*DAD*" MODEL

This chapter describes a general *DAD* model as a tri-level mixed-integer program (TLMIP). Direct solutions will typically be impossible, so we provide several indirect solution approaches.

We simplify model notation using the following conventions:

- Models and model instances are represented by acronyms in uppercase letters,
- A superscript indicates the type of inner optimization problem (e.g., "*LP*" for linear program or "*dLP*" for a dual linear program),
- Lowercase letters then identify the sense of each level's optimization, specifically, "*m*" for minimization and "*x*" for maximization.
- A "hat" over "*m*" or "*x*" indicates that the decision variables for that stage are fixed..

For example, $DAD^{LP}mxm$ stands for a tri-level defender-attacker-defender model with a min-max-min optimization structure, and with a linear program representing the defender-operator's optimization problem. And, $DAD^{LP}\hat{m}xm$ is really just a bi-level attacker-defender model because the defender's variables are fixed. (Appendix I contains the complete description of the notation used throughout this thesis.)

A. A GENERAL DEFENDER-ATTACKER-DEFENDER MODEL

1. Definitions and Model Assumptions

We state the inner "D problem" as

$$[\not{D}\not{A}D] \qquad \min_{\mathbf{y}\in Y(\mathbf{x})} f(\mathbf{y})$$

At this inner level of *DAD*, the defender-operator operates his system as best as possible by setting decision variables $\mathbf{y}$ to minimize operating cost, including penalties for unsatisfied constraints. This minimization can also represent other objectives such as maximizing operating profit or system output, minimizing unserved demand, and so

forth. The set $Y(\mathbf{x})$ represents operating constraints, e.g., flow-balance constraints in a pipeline model, as affected by a vector of attacks $\mathbf{x}$ that restricts that operation.

In fact, the attacker seeks maximize the defender-operator's cost of operating the damaged system, so the "AD model" is

$$[AD] \qquad \max_{\mathbf{x}\in X(\mathbf{w})} \min_{\mathbf{y}\in Y(\mathbf{x})} f(\mathbf{y}).$$

where $X(\mathbf{w})$ represents feasible attack plans after the defender implements a defensive plan $\mathbf{w}$. If component k is defended and made invulnerable, the assumption of transparency of information implies that component k will not be attacked. Of course, $X(\mathbf{w})$ will also include at least one resource constraint that limits the extent of possible attacks.

In the outer level of *DAD*, the defender uses his limited defensive resources to protect his system from attack. At this level, the defender's goal is to minimize the maximum damage that the attacker can inflict, where damage is measured in terms of the optimal solution to the defender-operator's inner model.

The three stages that this tri-level *DAD* model comprises are summarized as follows:

$$[DAD] \qquad \min_{\mathbf{w}\in W} \max_{\mathbf{x}\in X(\mathbf{w})} \min_{\mathbf{y}\in Y(\mathbf{x})} f(\mathbf{y})$$

The vector $\mathbf{w}$ represents the defensive actions taken by the defender to protect certain components of the system, and W represents the feasible region for the defender.

[*DAD*] posits that the defender wants to minimize the damage the attacker can cause. This will be accomplished by protecting some system components and, thereby, certain activities. For simplicity, we assume

1. Binary defensive actions: $w_k = 1$ if the k^{th} component has been protected, and $w_k = 0$ otherwise. The set W incorporates these binary restrictions as well as the defensive resource constraints.

2. Binary attacks: $x_k = 1$ if the k^{th} component k is attacked, and $x_k = 0$ otherwise. The set X incorporates these binary restrictions as well as the attacker's resource constraints.

3. Continuous activities: y_k represents the level of activity, set by the defender-operator, for component *k*. We assume that $\mathbf{0} \le \mathbf{y} \le \mathbf{u}$ and let $U = \text{diag}(\mathbf{u})$.

4. A "defense" completely armors a component. That is, $w_k = 1$ implies that component *k* is invulnerable to attack. Although $x_k = 1$ may be possible when $w_k = 1$, the attacker gains nothing from the corresponding attack. Since the attacker does not have so much resource that he can waste attacks on defended components, along with the assumption of perfect information, we may assume that $w_k = 1$ also implies $x_k = 0$.

5. One-to-one relationships are assumed between system components, attacks, and activities: A single attack stops exactly a single activity and an activity is stopped by no more or less than one attack.

The *DAD* model now becomes:

[*DAD*0] $$\min_{\mathbf{w} \in W} \quad \max_{\substack{\mathbf{x} \in X \\ \mathbf{x} \le \mathbf{1} - \mathbf{w}}} \quad \min_{\substack{\mathbf{y} \in Y \\ \mathbf{y} \le U(\mathbf{1} - \mathbf{x})}} \quad f(\mathbf{y}).$$

This model allows numerous generalizations such as "uninterdictable" activities and interdictions that affect more than one activity, although these generalizations will not be pursued in this thesis.

We also assume:

6. The property of *relatively complete recourse* prevails for *DAD* with respect to $\mathbf{w}$ and $\mathbf{x}$. (This property derives from the stochastic-programming literature; see Birge and Louveaux 1997, pp. 92-93). In particular, for any defensive plan $\mathbf{w} \in W$, the set $X(\mathbf{w}) = \{\mathbf{x} \in X \mid \mathbf{x} \le \mathbf{1} - \mathbf{w}\} \ne \varnothing$; and for any attack plan $\mathbf{x} \in X(\mathbf{w})$, the

set $Y(\mathbf{x}) = \{\mathbf{y} \in Y \mid \mathbf{y} \le U(\mathbf{1} - \mathbf{x})\} \neq \varnothing$. This means that, in all stages, the following player has a feasible response to the immediately preceding leader's play.

Proposition 1: For sufficiently large values of d_k, [*DAD*0] may be reformulated as

$$[DAD1] \qquad \min_{\mathbf{w} \in W} \max_{\mathbf{x} \in X} \min_{\mathbf{y} \in Y} f(\mathbf{y}) + (\mathbf{x}^T - \mathbf{w}^T)^+ D\mathbf{y},$$

where $\mathbf{d} = (d_1\, d_2 \ldots d_{|\mathcal{K}|})$, $D = diag(\mathbf{d})$, and the term $\left(\mathbf{x}^T - \mathbf{w}^T\right)^+$ stands for the vector maximum of $\mathbf{0}$ and $(\mathbf{x}^T - \mathbf{w}^T)$. ■

The proof is trivial. Actually, [*DAD*1] can also be used when interdiction of an activity does not force that activity's level to 0. For instance, suppose (a) the relevant CI system is the road network of a particular region; (b) the defender-operator wants to go from base A to airport B using a shortest path; (c) the attacker seeks to interdict road segments by means of bombardment (terrestrial, aerial, or via improvised explosive devices) to maximize the defender's shortest (quickest) A-B path; and (d) the defender can protect certain segments from attack with extra patrols or anti-aircraft weapons. In this case, an interdicted road segment might simply have a delay d_k added to its nominal traversal time c_k, and it may be worth the defender-operator's effort to incur this delay. We also note that the "+" operator here can be easily replaced by linear constructs and does not add any difficulty to the model's solution.

2. The Defender-Attacker-Defender Model [$DAD^{LP}mxm$]

When the inner minimization of [*DAD*1] is a linear program, the defender must solve this is problem:

$$[DAD^{LP}mxm] \quad \min_{\mathbf{w} \in W} \max_{\mathbf{x} \in X} \min_{\mathbf{y} \ge \mathbf{0}} \left(\mathbf{c}^T + \left(\mathbf{x}^T - \mathbf{w}^T\right)^+ D\right)\mathbf{y} \qquad (1)$$

$$\text{s.t.} \quad A^y\mathbf{y} = \mathbf{b}^y \quad [\boldsymbol{\pi}: \text{dual vars. for fixed } \mathbf{x} \text{ and } \mathbf{w}], \quad (2)$$

where the vector $\mathbf{c}$ denotes the activity costs and the constraints $A^y\mathbf{y} = \mathbf{b}^y$ correspond to general system-operation constraints. Note that, hereafter, dual variables for linear-programming restrictions of MIPs—for instance, $\boldsymbol{\pi}$ in $[DAD^{LP}mxm]$—will be denoted in square brackets next to the relevant constraints, but without the explanation as in (2).

Chapter IV proposes some solution procedures for $[DAD^{LP}mxm]$. However, some preliminary thoughts about how we might approach the problem will help up us to develop further models and solution procedures. The first attempt to solve this model might be to transform $[DAD^{LP}mxm]$ into $[DAD^{dLP}mxx]$. This is done by linearizing the expression in the objective function including the necessary constraints in the matrix A^y, temporarily fixing $\mathbf{w}$ and $\mathbf{x}$, taking the dual of the inner minimization problem, and releasing both variables after rearranging the terms. A bi-level mixed-integer linear program (BLMIP) results:

$$[DAD^{dLP}mxx] \quad \min_{\mathbf{w}\in W} \max_{\mathbf{x},\boldsymbol{\pi}} (\mathbf{b}^y)^T \boldsymbol{\pi} \tag{3}$$

$$\text{s.t. } (A^y)^T \boldsymbol{\pi} \le \mathbf{c} + D\left(\mathbf{x} - \mathbf{w}\right)^+ \tag{4}$$

$$\boldsymbol{\pi} \text{ free}, \ \mathbf{x} \in X \tag{5}$$

Given the last formulation, we would like again to take the dual of the inner maximization problem and solve a minimizing MIP problem by choice of $\mathbf{x}, \mathbf{y}, \mathbf{w}$. However, this is impossible because that inner maximization is not an LP. Therefore, we need an alternative approach.

We need bounds to enclose the objective function from above and below. Furthermore, when sequentially calculated in an iterative algorithm, these bounds must converge to the optimal value z^*.

Since the outer layer of the tri-level problem is a minimization, fixing the defense plan to $\mathbf{w} = \hat{\mathbf{w}}$, and solving the resulting problem, $[DAD^{LP}\hat{m}xm]$, leads to the upper bound $\overline{z}(\hat{\mathbf{w}}) \ge z^*$. By taking the dual of the inner minimization in $[DAD^{LP}\hat{m}xm]$, we see that it suffices to solve this MIP:

$$[DAD^{dLP}\hat{m}xx] \quad \overline{z}(\hat{\mathbf{w}}) = \max_{\mathbf{x},\boldsymbol{\pi}} \ (\mathbf{b}^y)^T \boldsymbol{\pi} \tag{6}$$

$$\text{s.t.} \ (A^y)^T \boldsymbol{\pi} \le \mathbf{c} + D\left(\mathbf{x} - \hat{\mathbf{w}}\right)^+ \tag{7}$$

$$\boldsymbol{\pi} \text{ free}, \quad \mathbf{x} \in X \tag{8}$$

This MIP can be solved either directly or by using a decomposition algorithm. A solution $\left(\hat{\mathbf{x}}, \hat{\boldsymbol{\pi}}\right)$ represents a tentative attack plan $\hat{\mathbf{x}}$ by the attacker and the resultant best dual response to that attack plan by the defender-operator. (Observe that the defender-operator's operating plan is obtained by solving the primal $[DAD^{LP}\hat{m}\hat{x}m]$ given both $\hat{\mathbf{w}}$ and $\hat{\mathbf{x}}$.)

Now, suppose that, for a given defensive plan $\hat{\mathbf{w}}$, we have enumerated all possible attack plans $\hat{\mathbf{x}}$ and corresponding extreme-point dual responses $\hat{\boldsymbol{\pi}}$ by the defender-operator. Let $(\hat{\mathbf{x}}, \hat{\boldsymbol{\pi}}) \in X\Pi$ denote this enumerated set, and let $\hat{X}\hat{\Pi}$ denote any nonempty subset of $X\Pi$. A master problem for the tri-level problem is defined as:

$$\begin{aligned} [MPm\hat{x}\hat{x}] \ \hat{z} = \ & \min_{z,\mathbf{w}} \quad z \\ & \text{s.t.} \quad z \ge (\mathbf{b}^y)^T \hat{\boldsymbol{\pi}} - \hat{\mathbf{x}}^T D\mathbf{w} \qquad \forall \ (\hat{\mathbf{x}}, \hat{\boldsymbol{\pi}}) \in \hat{X}\hat{\Pi} \end{aligned} \tag{9}$$

$$\mathbf{w} \in W \tag{10}$$

Observe that, whenever $\hat{x}_k = 1$, the term $-d_k w_k$ defines an upper bound on how much the defender would save if he had protected activity *k*.

Since the solution of the subproblem (equations (6)-(8)) certainly occurs at $\left(\hat{\mathbf{x}}, \hat{\boldsymbol{\pi}}\right) \in X\Pi$, it follows that $[MPm\hat{x}\hat{x}]$ is equivalent to $[DAD^{LP}mxx]$ when $\hat{X}\hat{\Pi} = X\Pi$. When $\hat{X}\hat{\Pi} \subseteq X\Pi$, we call $[MPm\hat{x}\hat{x}]$ the *relaxed master problem*. Indeed, it defines a relaxation of $[DAD^{dLP}mxx]$, and $\hat{z}$ gives a lower bound on the optimal objective to $[DAD^{LP}]$. Of course, we hope to obtain a solution by generating only a small subset $\hat{X}\hat{\Pi}$, with each $\left(\hat{\mathbf{x}}, \hat{\boldsymbol{\pi}}\right) \in \hat{X}\hat{\Pi}$ being generated in an iteration of a decomposition algorithm. The complete procedure is shown in Section IV.A.

B. A CAPACITY-INTERDICTION FORMULATION

One of the difficulties encountered in the formulation of [$DAD^{dLP}mxx$] is the apparent impossibility of transforming it into a simple minimization problem. Other ways to formulate and to solve the problem must be explored. This *capacity-interdiction* (CN) *model*, which assumes that the level of an interdicted activity must be 0, will prove useful:

$$[DAD^{LP}mxm\text{-CN1}] \quad \min_{\mathbf{w}\in W} \max_{\mathbf{x}\in X} \min_{\mathbf{y}\geq \mathbf{0}} \mathbf{c}^T\mathbf{y} \tag{11}$$

$$\text{s.t.} \quad A^y\mathbf{y} = \mathbf{b}^y \qquad [\boldsymbol{\alpha}] \tag{12}$$

$$\mathbf{y} \leq U(\mathbf{1} - \mathbf{x} + \mathbf{w}) \qquad [\boldsymbol{\beta}] \tag{13}$$

$$\mathbf{y} \leq \mathbf{u} \qquad [\boldsymbol{\theta}] \tag{14}$$

Essentially, this model is simply a reformulation of [$DAD0$] when the inner minimization is an LP.

Again, we may convert this problem into a bi-level nonlinear MIP (BLMINLP) by temporarily fixing $\mathbf{x}$, and taking the dual of the inner minimization problem. After rearranging terms and releasing $\mathbf{x}$, we obtain:

$$[DAD^{dLP}mxx\text{-CN1}] \quad \min_{\mathbf{w}\in W} \max_{\mathbf{x}\in X} \max_{\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\theta}} (\mathbf{b}^y)^T\boldsymbol{\alpha} + (\mathbf{1} - \mathbf{x} + \mathbf{w})^T U^T\boldsymbol{\beta} + \mathbf{u}^T\boldsymbol{\theta} \tag{15}$$

$$\text{s.t.} \quad (A^y)^T\boldsymbol{\alpha} + I\boldsymbol{\beta} + I\boldsymbol{\theta} \leq \mathbf{c} \tag{16}$$

$$\boldsymbol{\alpha} \text{ free}, \ \boldsymbol{\beta} \leq \mathbf{0}, \ \boldsymbol{\theta} \leq \mathbf{0} \tag{17}$$

This BLMINLP can be solved, at least in theory, by decomposition methods. (Note that a "max-max" is just a "max," so, in essence, the problem has been converted into a min-max defender-attacker model with a mixed-integer optimization model being solved by the attacker).

The next step to improve the model is to allow the defender-operator to make use of activities that have been interdicted (the penalties d_k are no longer sufficiently large to prevent that from happening). In order to do that, we need to slightly change the model to

$$[DAD^{LP}\,mxm\text{-CN2}] \quad \min_{\mathbf{w}\in W} \max_{\mathbf{x}\in X} \min_{\mathbf{y},\mathbf{y}'\geq \mathbf{0}} \mathbf{c}^T\mathbf{y} + (\mathbf{c}+\mathbf{d})^T\mathbf{y}' \tag{18}$$

$$\text{s.t.} \quad A^y\mathbf{y} + A^y\mathbf{y}' = \mathbf{b}^y \qquad [\boldsymbol{\alpha}] \tag{19}$$

$$\mathbf{y} \leq U(\mathbf{1} - \mathbf{x} + \mathbf{w}) \qquad [\boldsymbol{\beta}] \tag{20}$$

$$\mathbf{y} + \mathbf{y}' \leq \mathbf{u} \qquad [\boldsymbol{\theta}]. \tag{21}$$

Essentially, we are expanding the infrastructure by doubling the existing activities. By choosing y'_k instead of y_k, the defender makes use of the k^{th} activity, which has been interdicted, and he must, therefore, pay a penalty. Observe that the variable y'_k is not subject to the second set of constraints because it represents a fictitious activity that is neither interdicted nor defended and, by construction, it will only be used when the associated activity is interdicted.

Taking the dual of the inner min problem yields the following formulation:

$$[DAD^{dLP}\,mxx\text{-CN2}] \quad \min_{\mathbf{w}\in W} \max_{\mathbf{x}\in X} \max_{\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\theta}} (\mathbf{b}^y)^T\boldsymbol{\alpha} + (\mathbf{1} - \mathbf{x} + \mathbf{w})^T U^T\boldsymbol{\beta} + \mathbf{u}^T\boldsymbol{\theta} \tag{22}$$

$$\text{s.t.} \quad (A^y)^T\boldsymbol{\alpha} + I\boldsymbol{\beta} + I\boldsymbol{\theta} \leq \mathbf{c} \tag{23}$$

$$(A^y)^T\boldsymbol{\alpha} + I\boldsymbol{\theta} \leq \mathbf{c} + \mathbf{d} \tag{24}$$

$$\boldsymbol{\alpha} \text{ free}, \quad \boldsymbol{\beta} \leq \mathbf{0}, \ \boldsymbol{\theta} \leq \mathbf{0} \tag{25}$$

Chapter IV presents some methods to solve these models ($[DAD^{LP}]$ and $[DAD^{dLP}\text{-CN}]$) and their implementation for a specific problem: "Defending the Shortest Path" (DSP).

III. AN ATTACKER-DEFENDER-DEFENDER MODEL FOR BOUNDING "*DAD*"

This chapter formulates a tri-level model whose optimal objective value yields an optimistic bound for *DAD*. The bounding model reorders the stages in *DAD* to create an *ADD* model. This rearrangement simplifies the model, at least in theory, because the inner two minimization levels can now be collapsed into one. By means of "solution-elimination constraints," the bound yielded can be brought closer to the optimal *DAD* objective-function value to meet the stopping criterion of the decomposition method. Solution methods are proposed in Chapter IV.

A. MODEL FORMULATION

Let us consider $[DAD^{LP}mxm]$. Suppose that we interchange the first two levels by replacing min-max with max-min: The new model is $[ADD^{LP}xmm]$. Because the defender gets to observe the attacker's plan before making his own defensive decisions, which can nullify the effects of some individual attacks, we are giving the defender advantage. The optimal objective of such a model will yield a lower bound on the optimal objective value for $[DAD^{LP}mxm]$. This may be helpful for solving certain versions of this problem. The attacker-defender-defender model (*ADD*) may be formulated as follows:

$$[ADD^{LP}xmm] \quad \max_{\mathbf{x}\in X} \min_{\mathbf{w}\in W} \min_{\mathbf{y}\geq \mathbf{0}} \left(\mathbf{c}^T + \left(\mathbf{x}^T - \mathbf{w}^T\right)^+ D\right)\mathbf{y} \tag{26}$$

$$\text{s.t.} \quad A^y\mathbf{y} = \mathbf{b}^y \qquad [\boldsymbol{\pi}] \tag{27}$$

Alternatively, its compact form is:

$$[ADD^{LP}xm] \quad \max_{\mathbf{x}\in X} \min_{\mathbf{w}\in W,\ \mathbf{y}\geq \mathbf{0}} \left(\mathbf{c}^T + \left(\mathbf{x}^T - \mathbf{w}^T\right)^+ D\right)\mathbf{y} \tag{28}$$

$$\text{s.t.} \quad A^y\mathbf{y} = \mathbf{b}^y \qquad [\boldsymbol{\pi}] \tag{29}$$

As we have seen before, the inner minimization problem has a nonlinear objective function, and we cannot take its dual and obtain a linear, mixed-integer maximization problem. Thus, we may also want to use a *capacity-interdiction model* reformulation.

If we assume that the penalties d_k are much greater than the cost of the activities, and that $w_k = 0$ and $x_k = 1$ imply that $y_k = 0$, we can rewrite $[ADD^{LP}xmm]$ as:

$$[ADD^{LP}xmm\text{-CN1}] \quad \max_{\mathbf{x}\in X} \min_{\mathbf{w}\in W,\mathbf{y}\geq\mathbf{0}} \quad \mathbf{c}^T\mathbf{y} \tag{30}$$

$$\text{s.t.} \quad A^y\mathbf{y} = \mathbf{b}^y \qquad [\boldsymbol{\alpha}] \tag{31}$$

$$\mathbf{y} \leq U(\mathbf{1} - \mathbf{x} + \mathbf{w}) \qquad [\boldsymbol{\beta}] \tag{32}$$

$$\mathbf{y} \leq \mathbf{u} \qquad [\boldsymbol{\theta}] \tag{33}$$

Constraints (32) establish an upper bound on activities depending upon interdiction and defense. Constraints (33) represent capacity limitations for every component (an example is maximum flow across a pipe segment of a water system). Note that we invoke the property of "relatively complete recourse" here to ensure that the model is feasible for any feasible attack plan $\mathbf{x}$.

If we do not make the assumption that the penalties d_k are large enough to preclude the utilization of interdicted activities, we may enhance the model as follows:

$$[ADD^{LP}xmm\text{-CN2}] \quad \max_{\mathbf{x}\in X} \min_{\mathbf{w}\in W,\mathbf{y},\mathbf{y}'} \quad \mathbf{c}^T\mathbf{y} + (\mathbf{c}+\mathbf{d})^T\mathbf{y}' \tag{34}$$

$$\text{s.t.} \quad A^y\mathbf{y} + A^y\mathbf{y}' = \mathbf{b}^y \qquad [\boldsymbol{\alpha}] \tag{35}$$

$$\mathbf{y} \leq U(\mathbf{1} - \mathbf{x} + \mathbf{w}) \qquad [\boldsymbol{\beta}] \tag{36}$$

$$\mathbf{y} + \mathbf{y}' \leq \mathbf{u} \qquad [\boldsymbol{\theta}] \tag{37}$$

$$\mathbf{y} \geq \mathbf{0}, \mathbf{y}' \geq \mathbf{0} \tag{38}$$

Recall that by selecting y'_k instead of y_k, the defender agrees to use the k^{th} activity with the added penalty.

In Chapter IV, we develop some methods to solve these models.

B. STRONGER BOUNDS: STRENGTHENING "*ADD*"

Although the solution to *ADD* gives a lower bound for *DAD*, the bound may be poor. Assume, for instance, that *DAD* has simple cardinality constraints for attacker and defender resources and the right-hand sides of those constraints are the same. In this case, the solution $(\hat{\mathbf{w}}, \hat{\mathbf{y}})$ obtained from $[ADD^{LP}\hat{x}mm]$ for any fixed interdiction plan $\hat{\mathbf{x}}$, will

nullify all the attacks implied by $\hat{\mathbf{x}}$. Thus, the bound is the obviously weak bound provided by the solution to the defender-operator's problem assuming no interdictions at all. We can do better.

Consider the directed network depicted in Figure 1 where the defender-operator's objective is to find the shortest path from *s* to *t*. The nominal length of each arc (i, j) is $c_{i,j}$, and the potential delay that the attacker can inflict is $d_{i,j}$. The attacker will attack two arcs and the defender will defend one. The optimal objective value for *DAD* is 3, but the lower bound coming from *ADD* is 2 (Figure b). If we give three units of resource to the attacker, the optimal objective value for the new problem, denoted by "ADD^+," is 3 (Figure c) and, therefore, it provides a valid and stronger lower bound on the optimal objective-function value z^*.

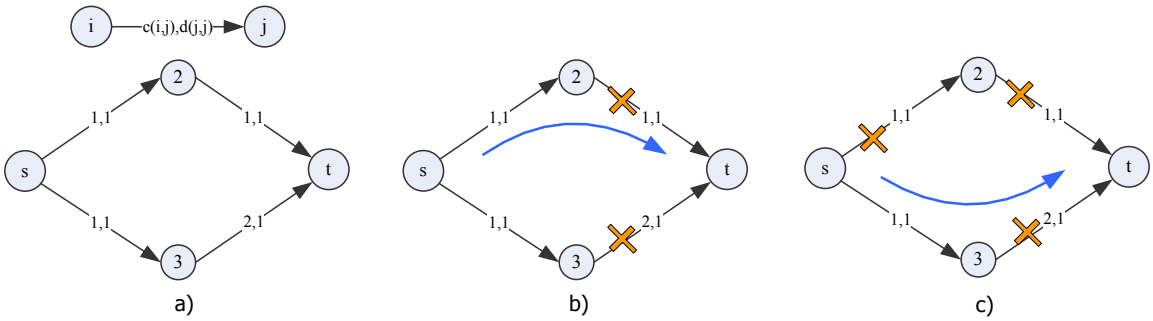

Figure 1. Network to illustrate the tightening and validity of the ADD^+ lower bound. Figure a) is the original network; figure b) depicts the solution to *ADD* and figure c) depicts the solution to ADD^+.

Thus, to tighten the lower bound from *ADD*, we may try to give the attacker extra resource. However, if we give him too much, the resulting bound may be not valid. So, how much extra resource can we give to the attacker and still be sure of a valid bound?

As in the example, assume that limits on resources for the attacker and the defender are given by simple knapsack constraints, i.e., $\mathbf{x} \in X \equiv \left\{ \mathbf{x} \in \{0,1\}^n \mid \mathbf{a}^x \mathbf{x} \le b^x \right\}$ and $\mathbf{w} \in W \equiv \left\{ \mathbf{w} \in \{0,1\}^n \mid \mathbf{a}^w \mathbf{w} \le b^w \right\}$, respectively. Now, create an instance of ADD^+ by

giving the attacker $\delta \geq 0$ extra units of resource, i.e., replace $\mathbf{x} \in X$ in *ADD* with $\mathbf{x} \in X(\delta) \equiv \left\{ \mathbf{x} \in \{0,1\}^n \mid \mathbf{a}^x \mathbf{x} \leq b^x + \delta \right\}$. The model is formulated as follows:

$$[ADD^{+\,LP}xmm(\delta)] \quad \max_{\mathbf{x} \in X(\delta)} \min_{\mathbf{w} \in W} \min_{\mathbf{y} \geq \mathbf{0}} \left(\mathbf{c}^T + \left(\mathbf{x}^T - \mathbf{w}^T \right)^+ D \right) \mathbf{y}$$

$$\text{s.t.} \quad A^y \mathbf{y} = \mathbf{b}^y \qquad [\boldsymbol{\pi}]$$

where δ would be the least interdiction resource which would make the defender use a maximal defensive resource to counter it. A valid value for δ still needs to be found.

Theorem 1: Let z^* be the optimal objective value of $[DAD^{LP}mxm]$, let $\underline{z}^*(\delta)$ be the optimal objective value for $[ADD^{+\,LP}xmm(\delta)]$, and define $\overline{W} \equiv W \cap \left\{ \mathbf{w} \text{ is maximal for } \mathbf{a}^w \mathbf{w} \leq b^w \right\}$. If

$$\delta \leq \min_{\mathbf{w} \in \overline{W}} \mathbf{a}^x \mathbf{w}, \tag{39}$$

then $\underline{z}^*(\delta) \leq z^*$.

Proof: Let us write $[DAD^{LP}mxm]$ with knapsack constraints for the defender and attacker, and a general linear program for the defender-operator in this simplified form of equation (1):

$$z^* = \min_{\mathbf{w} \in W} \max_{\mathbf{x} \in X} f(\mathbf{x}, \mathbf{w}) \tag{40}$$

where $f(\mathbf{x}, \mathbf{w}) \equiv \min_{\mathbf{y} \in Y} \left(\mathbf{c}^T + \left(\mathbf{x}^T - \mathbf{w}^T \right)^+ D \right) \mathbf{y}$.

Now suppose that the attacker has δ extra units of attack resource, with δ satisfying (39), but must waste that resource on defended activities. This will have no effect in the objective-function value because it is always feasible for the defender to neutralize the "extra" attacks. Therefore,

$$z^* = \min_{\mathbf{w} \in W} \max_{\mathbf{x} \in X(\delta)} f(\mathbf{x}, \mathbf{w}) \tag{41}$$

$$\text{s.t.} \sum_k a_k^x x_k w_k \geq \delta \tag{42}$$

Since the attacker must act first, interchanging the max and min benefits the defender. Thus,

$$z^* \geq \max_{\mathbf{x} \in X(\delta)} \min_{\mathbf{w} \in W} f(\mathbf{x}, \mathbf{w}) \tag{43}$$

$$\text{s.t.} \sum_k a_k^x x_k w_k \geq \delta \tag{44}$$

$$= \max_{\mathbf{x} \in X(\delta)} \min_{\mathbf{w} \in W} f(\mathbf{x}, \mathbf{w}) \tag{45}$$

$$= \underline{z}^*(\delta) \qquad \text{by definition.}$$

Equality holds in (45) because (a) we may assume the defender, who now plays second, will "post-defend" only interdicted activities and, thus, the left-hand side of (44) will always be positive; and (b) constraint (39) ensures that that positive left-hand side is always at least δ.

It may be the case that, even if δ does not satisfy (39), $\underline{z}^*(\delta)$ is still a valid lower bound on z^*. For example, in the simple *DAD* problem described in Figure 1, $\delta = 1$ is the maximum value that is guaranteed to be valid by Theorem 1. However, for $\delta = 2$, $\underline{z}^*(\delta) = 3$ is still a valid lower bound (in this case, $\underline{z}^*(\delta) = z^*$). However, "cheating" in this way may not always be possible.

The solution of ADD^+ must be a feasible solution in *DAD* to be of any value. Therefore, the defender's actions must not only be feasible with respect to the initial constraints W, but also need to nullify some of the attacks. The remaining ones either represent a feasible solution for the attacker in the original *DAD*, or they do not have an impact the objective-function value. Consequently, the defensive constraints in ADD^+ are given by $\overline{W}$.

To summarize the last two paragraphs more precisely, the solution of ADD^+ will give a lower bound for *DAD* provided that, in *DAD*, the optimal objective value is non-increasing for increasing $\mathbf{x}$ (in *A*), and non-decreasing for increasing $\mathbf{w}$ (in *D*). That is,

extra attacks (δ) do not consume extra defensive resource because they do not affect the objective-function value. Neither do extra defenses added to the problem, because the best operating plan has already been protected and at least b^x attacks have been left uncovered. In this sense, the proof requires that $\mathbf{w}$ be "maximal" with respect to constraints $\overline{W}$.

For the special case where both the attacker and defender have cardinality constraints in their respective resources, we can think of the following game sequence in ADD^+:

- The attacker interdicts $b^x + b^w$ activities;
- The defender nullifies b^w of those attacks and brings back the associated activities to their original costs; and
- Finally, the defender-operator again finds the optimal operational plan for the system given the increased costs for non-nullified attacks.

Ideally, ADD^+ is solved in hopes that the solution yielded ($\mathbf{w}^*_{ADD^+}$) is close enough to the *DAD* objective-function value to meet optimality criterion. Thus, the optimality conditions need to be tested in the *DAD* subproblem formulated in equations (6)-(8) having $\mathbf{w}^*_{ADD^+}$ as an input $\left(\hat{\mathbf{w}}_{DAD} \leftarrow \mathbf{w}^*_{ADD^+}\right)$. If the upper bound of [$DAD^{dLP}\hat{m}xx$] reveals a non-optimal gap, that is, $\overline{z}_{DAD} - \underline{z}_{ADD} > \varepsilon$, then ADD^+ has to be recalculated with an added solution-elimination constraint $\mathbf{w} \neq \hat{\mathbf{w}}_{ADD^+}$. Hopefully, the convergence of this algorithm to get an ε-optimal solution for *DAD* is faster than the decomposition method for *DAD* itself.

The next chapter provides some statistics about the bound's quality that this ADD^+ model yields.

IV. SOLUTION METHODOLOGIES

Thus far, we have presented variations on *DAD* and models for bounding *DAD*. This section adds detail and illustrates general procedures for solving these models.

Section A provides a basic algorithm to solve *DAD* and illustrates this solution procedure by solving the problem of "Defending the Shortest Path" (DSP). Section B solves DSP using the capacity-interdiction version of *DAD*. Sections C and D deal with *ADD* and ADD^{+} and build a special model (*MXSP*) for solving DSP. This is derived from ADD^{+} and is based on a network-interdiction problem on an expanded network with a particular structure.

A. DEFENDER-ATTACKER-DEFENDER MODEL

Chapter II sketched an algorithm to solve *DAD* by decomposition. We agree that any feasible defense plan would give us an upper bound on the objective function. This bound can be computed by fixing $\mathbf{w} = \hat{\mathbf{w}}$ and solving the following "upper-bounding subproblem:"

[$DAD^{dLP}\hat{m}xx$-SP]

$$\begin{aligned} \overline{z}(\hat{\mathbf{w}}) = \max_{\mathbf{x},\boldsymbol{\pi}} \quad & (\mathbf{b}^{y})^{T}\boldsymbol{\pi} \\ \text{s.t.} \quad & (A^{y})^{T}\boldsymbol{\pi} \le \mathbf{c} + D\left(\mathbf{x} - \hat{\mathbf{w}}\right)^{+} \\ & \boldsymbol{\pi} \text{ free} \\ & \mathbf{x} \in X. \end{aligned}$$

[$DAD^{dLP}\hat{m}xx$-SP] is a mixed-integer program. We may try to solve it either directly or by decomposition. In this latter case we would have an "inner" Benders-like decomposition method for the subproblem, and an "outer" decomposition for the full *DAD* (Henceforth we will call this method "nested decomposition," O'Neill 1976). The decomposition for the subproblem is defined by these two problems:

$$[SP'\hat{m}\hat{x}m] \qquad \min_{\mathbf{y}\geq\mathbf{0}} \quad \left(\mathbf{c}^T + \left(\hat{\mathbf{x}}^T - \hat{\mathbf{w}}^T\right)^+ D\right)\mathbf{y} \tag{46}$$

$$\text{s.t.} \quad A^y\mathbf{y} = \mathbf{b}^y \tag{47}$$

$$[MP'\hat{m}x\hat{m}] \qquad \max_{z,\mathbf{x}} \quad z \tag{48}$$

$$\text{s.t.} \quad z \leq \mathbf{c}^T\hat{\mathbf{y}} + \left(\mathbf{x}^T - \hat{\mathbf{w}}^T\right)^+ D\hat{\mathbf{y}}, \forall\,(\hat{\mathbf{w}},\hat{\mathbf{y}}) \in \hat{W}\hat{Y} \tag{49}$$

$$\mathbf{x} \in X, \tag{50}$$

where the set $\hat{W}\hat{Y}$ in (48) comprises all pairs $(\hat{\mathbf{w}},\hat{\mathbf{y}})$ identified by the subproblem (46) on successive iterations.

Now, $[SP'\hat{m}\hat{x}m]$ is an LP and $[MP'\hat{m}x\hat{m}]$ is just a MIP. The (relaxed) master problem is defined as usual:

$$[DAD^{dLP}m\hat{x}\hat{x}\text{-}MP]$$

$$\underline{z}(\hat{X}\,\hat{\Pi}) = \min_{z,\mathbf{w}} \quad z \tag{51}$$

$$\text{s.t.} \quad z \geq (\mathbf{b}^y)^T\hat{\boldsymbol{\pi}} - \hat{\mathbf{x}}^T D\mathbf{w} \qquad \forall\,(\hat{\mathbf{x}},\hat{\boldsymbol{\pi}}) \in \hat{X}\,\hat{\Pi} \tag{52}$$

$$\mathbf{w} \in W \tag{53}$$

where $(\hat{\boldsymbol{\pi}},\hat{\mathbf{x}})$ come from the solution of the subproblem either directly or by decomposition. In theory, the algorithm would eventually enumerate all possible feasible combinations for $(\boldsymbol{\pi},\mathbf{x},\mathbf{w})$, so a solution must be found. Let us outline this algorithm.

1. A Decomposition Algorithm to Solve *DAD*

Algorithm 1:

Input: An instance of $[DAD^{LP}mxm]$ with matrices D, A^y, initial feasible defense plan, $\hat{\mathbf{w}}^0$ (e.g., $\hat{\mathbf{w}}^0 = \mathbf{0}$), vectors $\mathbf{b}^y$ and $\mathbf{c}$, and an allowable optimality gap ε.

Output: An ε-optimal defensive plan $\mathbf{w}^*$ for $[DAD^{LP}mxm]$, the optimal attack plan $\mathbf{x}^*$, and the optimal system-operation plan $\mathbf{y}^*$.

{

Initialize: $\overline{z} \leftarrow \infty$; $\underline{z} \leftarrow -\infty$; $\hat{\mathbf{w}} \leftarrow \hat{\mathbf{w}}^0$; $\hat{X}\,\hat{\Pi} \leftarrow \varnothing$

While $\left(\overline{z} - \underline{z} > \varepsilon\right)$ {

Solve [$DAD^{dLP}\hat{m}xx$-SP] for $\hat{\mathbf{w}}$ (either directly or by decomposition) to obtain an incumbent upper bound on the objective function $\overline{z}(\hat{\mathbf{w}})$; $\hat{X}\hat{\Pi} \leftarrow \hat{X}\hat{\Pi} \cup \{(\hat{\mathbf{x}}, \hat{\boldsymbol{\pi}})\}$;

If $\left(\overline{z}(\hat{\mathbf{w}}) < \overline{z}\right)$ { $\overline{z} \leftarrow \overline{z}(\hat{\mathbf{w}})$; $\hat{\mathbf{w}}^* \leftarrow \hat{\mathbf{w}}$; }

Solve [$DAD^{dLP}m\hat{x}\hat{x}$-MP] for all $(\hat{\mathbf{x}}, \hat{\boldsymbol{\pi}}) \in \hat{X}\hat{\Pi}$ to obtain $\underline{z}(\hat{X}\hat{\Pi})$ and a new defense plan $\hat{\mathbf{w}}$;

$\underline{z} \leftarrow \underline{z}(\hat{X}\hat{\Pi})$;

}

Print ("ε-optimal defense plan, activity levels and objective-function values are" $\mathbf{w}^*$, $\mathbf{y}^*$, z^*, "respectively.")

}

The nested decomposition algorithm (Algorithm 1A) is identical to Algorithm 1. The only difference is that the subproblem is solved inside an inner loop that takes $\hat{\mathbf{w}}$ as a fixed parameter and, then, proceeds to solve the sub-subproblem [$SP'\hat{m}\hat{x}m$] in (46)-(47) This is followed by the sub-master problem [$MP'\hat{m}x\hat{m}$] in (48)-(50). The solution obtained from this inner decomposition loop $\left(\hat{\mathbf{x}}, \hat{\mathbf{y}}\right)$ is now injected into the outer master problem. Then, and Algorithm 1 takes over.

Infanger and Morton (1996) propose sharing cuts for different scenarios at the same stage in order to accelerate the convergence of decomposition methods to solve multi-stage stochastic linear programs. In the same fashion, it may be possible to use results obtained in the solution of the subproblem [$DAD^{dLP}\hat{m}xx$-SP] from one major iteration to the next. This may improve substantially the solution times reported in Table 9, but is beyond the scope of this thesis.

2. Implementation of *DAD* for "Defending the Shortest Path"

We will illustrate Algorithm 1 with the *DAD* problem "Defending the Shortest Path" (*DSP*). In *DSP*, the defender needs to minimize the maximum traversal length of the network that represents the system infrastructure. We might be dealing with, for example, a railway transportation system, a military logistic depot at station A, a potential customer at station B, and a certain commodity that must be sent from A to B in the shortest time possible.

The system is modeled as a network with its corresponding set of nodes (e.g., train stations on a railway system or road intersections on a road network) and arcs (e.g., segments of railways connecting stations or road segments between intersections). Costs are defined by arc-traversal times and penalties are defined by delays incurred if interdicted arcs are traversed.

For this model and other models implemented in this thesis, we assume that action occurs on arcs. If we wish to take action over a node in the network, we just split it and propagate the interdiction and defense through all the arcs that connect the split nodes (Ahuja, Magnati and Orlin 1993, pp. 41-42). Furthermore, for simplicity, we consider only cardinality constraints in the resources for both attacker and defender.

Problem definition: Minimize the maximum traversal cost that the attacker is able to inflict after his best attack by appropriately selecting the arcs to be protected subject to available resources. The system is represented by the directed graph $\mathcal{G}==(\mathcal{N},\mathcal{A})$ which contains a set of nodes $i \in \mathcal{N}$ and the linking arcs $k=(i,j) \in \mathcal{A}$ with their cost c_k and penalty d_k. $\mathcal{A}$ represents the set of all arcs and $\mathcal{N}$ represents the set of all nodes. If an arc k is traversed, the defender pays its nominal cost c_k. However, if that arc has been interdicted, an extra cost d_k is added The latter is to be applied if the defender-operator traverses k.

Indices and index sets:

$i, j \in \mathcal{N}$ Nodes in $\mathcal{G}=(\mathcal{N},\mathcal{A})$

s Source node in $\mathcal{G}=(\mathcal{N},\mathcal{A})$

t Sink node in $\mathcal{G}=(\mathcal{N},\mathcal{A})$

$k=(i,j)\in\mathcal{A}$ Arcs in $\mathcal{G}=(\mathcal{N},\mathcal{A})$ connecting nodes $i,j\in\mathcal{N}$

$\mathcal{FS}(i)$ Forward Star of node i. (Set of all arcs $k\in\mathcal{A}$ departing from i)

$\mathcal{RS}(i)$ Reverse Star of node i. (Set of all arcs $k\in\mathcal{A}$ arriving at i)

Data:

$c_k\geq 0$ Nominal cost of traversing arc k

$d_k\geq 0$ Added penalty the defender-operator pays if arc k is interdicted and then, traversed

b^x Maximum number of attacks to the network (attacker's resource)

b^w Maximum number of arcs that can be protected (defender resource)

Variables:

x_k Attacker's decision to interdict arc k: $x_k=1$ if arc k is interdicted, and $x_k=0$ otherwise

y_k Defender's decision to traverse arc k which has not been interdicted: $y_k=1$ if arc k is traversed, and $y_k=0$ otherwise

π_i Dual variables for flow-balance constraints at each $i\in\mathcal{N}$

w_k Defender's decision to defend arc k: $w_k=1$ if arc k is defended, and $w_k=0$ otherwise

Formulation of the basic DSP model:

$[DAD^{LP}mxm]$

$$\min_{\mathbf{w}\in W_{DSP}}\max_{\mathbf{x}\in X_{DSP}}\min_{\mathbf{y}} \quad \sum_{k\in\mathcal{A}}\left(c_k+d_k x_k\left(1-w_k\right)\right)y_k \tag{54}$$

$$\text{s.t.} \sum_{k \in \mathcal{FS}(i)} y_k - \sum_{k \in \mathcal{RS}(i)} y_k = \begin{cases} 1 & \text{if } i = s \\ 0 & \forall i \in \mathcal{N} \setminus \{s,t\} \\ -1 & \text{if } i = t, \end{cases} \tag{55}$$

where

$$X_{DSP} = \left\{ \mathbf{x} \in \{0,1\}^{|\Omega|} \mid \sum_{k \in \mathcal{A}} x_k \le b^x \right\}, \text{ and} \tag{56}$$

$$W_{DSP} = \left\{ \mathbf{w} \in \{0,1\}^{|\Omega|} \mid \sum_{k \in \mathcal{A}} w_k \le b^w \right\}. \tag{57}$$

Formulation of the subproblem for DSP:

Since we are using decomposition, the following subproblem (see $[DAD^{dLP}\hat{m}xx]$, equations (3)-(5)) must be solved:

$[DAD^{dLP}\hat{m}xx\text{-SP}]$

$$\bar{z}(\hat{\mathbf{w}}) = \max_{\boldsymbol{\pi}, \mathbf{x}} \quad \pi_s - \pi_t \tag{58}$$

$$\text{s.t}: \quad \pi_j - \pi_i \le c_k + d_k x_k (1 - \hat{w}_k) \qquad \forall k \in \mathcal{A} \tag{59}$$

$$\mathbf{x} \in X_{DSP} \tag{60}$$

$$\pi_i \text{ free} \quad \forall i \in \mathcal{N}$$

Formulation of the master problem for DSP:

See $[DAD^{dLP}m\hat{x}\hat{x}\text{-}MP]$, equations (51)-(53).

$[DAD^{dLP}m\hat{x}\hat{x} - MP]$

$$\underline{z}(\hat{X}\,\hat{\Pi}) = \min_{\mathbf{w}, z} \quad z \tag{61}$$

$$\text{s.t.} \quad z \ge \left(\hat{\pi}_t - \hat{\pi}_s\right) - \sum_{k \in \mathcal{A}} \hat{x}_k d_k w_k \quad \forall (\hat{\mathbf{x}}, \hat{\boldsymbol{\pi}}) \in \hat{X}\,\hat{\Pi} \tag{62}$$

$$\mathbf{w} \in W_{DSP} \tag{63}$$

Constraints (59) are the optimality conditions for the DSP and constraint (60) is the attacker's resource constraint, along with the integrality requirements for the

variables. Constraints (62) are the Benders cuts for each iteration of the algorithm, and constraint (63) is the defender's resource constraint, along with the integrality requirements for the variables.

Chapter V provides some computational results using this algorithm on grid networks of different aspects and sizes.

3. Stronger Bounds for *DAD*

The lower bound provided by the master problem [$DAD^{dLP}m\hat{x}\hat{x}$-MP] in equations (51)-(53) can be tightened by solving a linear program that calculates the best remaining operating plan after the defender utilizes defenses to nullify some of the attacks.

Let us illustrate with a simple example. Consider the DSP depicted in Figure 2, where the defender, with 1 unit of defensive resource, needs to traverse from node 1 to node 4. The attacker, with 2 units of offensive resource, wants to maximize the length of the defender-operator's route.

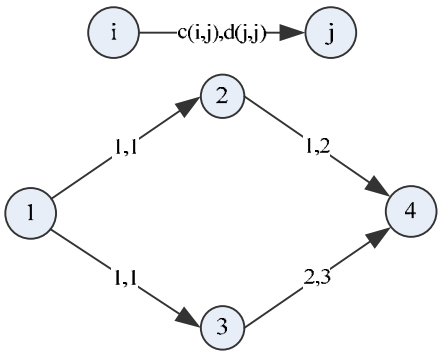

Figure 2. 4-node network to show a simple DSP problem where the lower bound provided by the master problem can be tightened.

In the subproblem's first iteration, because the attacker chooses to attack $\hat{\mathbf{x}} = \{(2,4),(3,4)\}$ and the defender-operator chooses to traverse $\hat{\mathbf{y}} = \{(1,2),(2,4)\}$, the upper bound is set at $\overline{z} = 4$. Then, the master problem suggests defending $\hat{\mathbf{w}} = \{(3,4)\}$ and, because 3 units of penalty $\left(d_{3,4} = 3\right)$ are subtracted from the current objective value, the bound is brought down to $\underline{z} = 1$. However, given the last attacks, the defender could

have done better by defending $\hat{\mathbf{w}} = \{(2,4)\}$ This would increase the lower bound to $\underline{z} = 2$ when the defender traverses $\hat{\mathbf{y}} = \{(1,2),(2,4)\}$.

Observe that when the defender has more resources than the attacker, this bound will always equal the uninterdicted shortest-path length.

A valid lower bound can be obtained from the solution of the following *capacity-expansion* LP model, specialized for the DSP.

$$[LP\hat{m}\hat{x}m] \qquad \min_{\mathbf{y},\mathbf{y}' \geq \mathbf{0}} \quad \mathbf{c}'^{T}\mathbf{y} + \hat{\mathbf{x}}^{T} C\mathbf{y}' \tag{64}$$

$$\text{s.t.} \quad A^{y}\mathbf{y} + A^{y}\mathbf{y}' = \mathbf{b}^{y} \tag{65}$$

$$\hat{\mathbf{x}}^{T}\mathbf{y}' \leq b^{w} \tag{66}$$

$$(\mathbf{1} - \hat{\mathbf{x}}^{T})\mathbf{y}' = 0 \tag{67}$$

$$\mathbf{y} + \mathbf{y}' \leq \mathbf{1}, \tag{68}$$

where $D = diag(\mathbf{d})$, $C = diag(\mathbf{c})$ and $\mathbf{c}' = (1 - \hat{\mathbf{x}})C + \hat{\mathbf{x}}(C + D)$. The vector $\mathbf{y}'$ denotes the defender's decision to protect and traverse an interdicted set of arcs and $\mathbf{y}$ denotes the defender's decision to traverse and take no defensive action on a different set of arcs, which may or may not have been interdicted. Constraint (66) represents an upper limit in the number of interdicted activities that may be protected and then, traversed. Constraint (67) restricts the use of $\mathbf{y}'$ to those activities that have been attacked. Finally, constraints (68) are the capacity expansion constraints for every activity.

For the example presented above, the system transformation implied by the capacity-expansion LP model is depicted in the following figure:

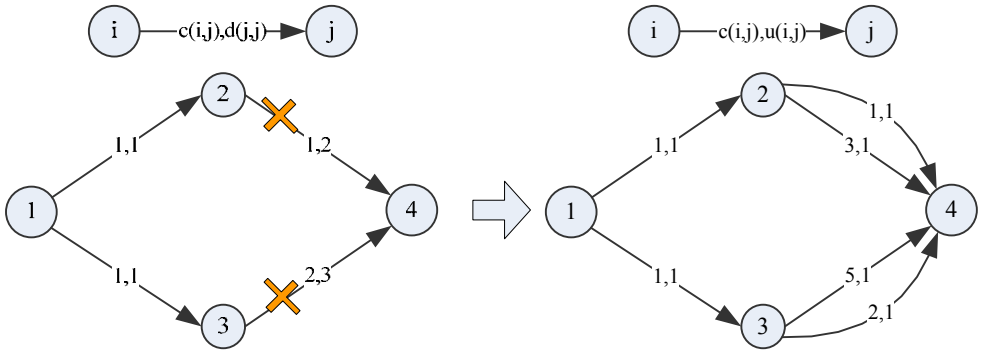

Figure 3. Transformation of the interdicted system of Figure 1 to solve a capacity-expansion model and obtain a lower bound. The interdicted activities are "doubled" with a total cost of $c'_k = c_k + d_k$. Observe that the arcs are labeled in terms of cost and capacity instead of cost and penalty.

The defender must choose a path from 1 to 4 that minimizes the cost and that implies the selection of, at most, one of the newly created arcs.

Algorithm 1B is a modification of Algorithm 1 where, right after de master problem, the capacity-expansion model is solved for the previous pair of $\hat{\mathbf{w}}$ and $\hat{\mathbf{x}}$. In the computational chapter, we shall see the tradeoffs between adding an extra step in Algorithm 1B to solve an LP, and more importantly, the improvement attained by tightening the lower bound as well as the validity of this bound.

B. CAPACITY-INTERDICTION DEFENDER-ATTACKER-DEFENDER MODEL

This model is introduced in formulas (11)-(14) and is repeated here for reference. It is a bi-level MIP, which is, again, difficult to solve directly:

$$[DAD^{LP}mxm\text{-CN1}] \quad \min_{\mathbf{w}\in W} \max_{\mathbf{x}\in X} \min_{\mathbf{y}\geq \mathbf{0}} \mathbf{c}^T\mathbf{y}$$

$$\text{s.t.} \quad A^y\mathbf{y} = \mathbf{b}^y \qquad [\boldsymbol{\alpha}]$$
$$\mathbf{y} \leq U(\mathbf{1} - \mathbf{x} + \mathbf{w}) \qquad [\boldsymbol{\beta}]$$
$$\mathbf{y} \leq \mathbf{u} \qquad [\boldsymbol{\theta}]$$

As we claimed at the beginning of the previous chapter, any feasible $\hat{\mathbf{w}}$ will eventually determine an upper bound $\overline{z} \geq z^*$. We need to solve the Benders subproblem [$DAD^{LP}\hat{m}xx$] for $\hat{\mathbf{w}}$:

[$DAD^{dLP}\hat{m}xx$-CN-SP]

$$\overline{z}(\hat{\mathbf{w}}) = \max_{\mathbf{x},\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\theta}} \quad (\mathbf{b}^y)^T\boldsymbol{\alpha} + (\mathbf{1} - \mathbf{x} + \hat{\mathbf{w}})^T\boldsymbol{\beta} + \mathbf{u}^T\boldsymbol{\theta} \tag{69}$$

$$\text{s.t.} \quad (A^y)^T\boldsymbol{\alpha} + I\boldsymbol{\beta} + I\boldsymbol{\theta} \leq \mathbf{c} \tag{70}$$

$$\boldsymbol{\alpha} \text{ free}, \quad \boldsymbol{\beta} \leq \mathbf{0}, \ \boldsymbol{\theta} \leq \mathbf{0} \tag{71}$$

$$\mathbf{x} \in X \tag{72}$$

However, the objective function is still non-linear in the term $\left(\mathbf{1} - \mathbf{x} + \hat{\mathbf{w}}\right)^T \boldsymbol{\beta}$. Since $\hat{\mathbf{w}}$ is fixed beforehand, we can break up the problem for the different values that $\hat{w}_k$ can take on, either 1 or 0:

- When $\hat{w}_k = 1$, the k^{th} activity cannot be attacked and the term $\left(\mathbf{1} - \mathbf{x} + \hat{\mathbf{w}}\right)^T \boldsymbol{\beta}$ becomes $\sum_{k|\hat{w}_k=1} 2\beta_k$

- When $\hat{w}_k = 0$ we have $\left(\mathbf{1} - \mathbf{x} + \hat{\mathbf{w}}\right)^T \boldsymbol{\beta} = \sum_{k|\hat{w}_k=0} \left(1 - x_k\right)\beta_k$, which is still non-linear. Extra variables and constraints are needed to linearize the term in this case:

$$\begin{aligned} \sum_{k|\hat{w}_k=0} \left(1 - x_k\right)\beta_k \quad &= \quad \sum_{k|\hat{w}_k=0} \beta'_k \\ \text{s.t.} \quad & \beta'_k \geq -M\left(1 - x_k\right) && \forall\, k \mid \hat{w}_k = 0 \\ & \beta'_k \leq \beta_k + Mx_k && \forall\, k \mid \hat{w}_k = 0 \\ & \beta'_k \leq 0 && \forall\, k, \end{aligned}$$

where M is a sufficiently large number so that the first inequality is obviated whenever $x_k = 0$.

The subproblem can be rewritten as follows:

[$DAD^{dLP}\hat{m}xx$-CN-SP]

$$\overline{z}(\hat{\mathbf{w}}) = \max_{\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\theta},\mathbf{x}} (\mathbf{b}^y)^T\boldsymbol{\alpha} + \sum_{k|\hat{w}_k=1} 2\beta_k + \sum_{k|\hat{w}_k=0} \beta'_k + \mathbf{u}^T\boldsymbol{\theta} \tag{73}$$

$$\text{s.t.} \quad (A^y)^T\boldsymbol{\alpha} + I\boldsymbol{\beta} + I\boldsymbol{\theta} \le \mathbf{c}$$

$$\beta'_k \ge -M\left(1-x_k\right) \quad \forall k \mid \hat{w}_k = 0 \tag{74}$$

$$\beta'_k \le \beta_k + Mx_k \quad \forall k \mid \hat{w}_k = 0 \tag{75}$$

$$\beta'_k \le 0 \quad \forall k \tag{76}$$

$$\boldsymbol{\alpha} \text{ free}, \ \boldsymbol{\beta} \le \mathbf{0}, \boldsymbol{\beta}' \le \mathbf{0}, \ \boldsymbol{\theta} \le \mathbf{0}$$

$$\mathbf{x} \in X$$

With solutions $\left(\hat{\boldsymbol{\alpha}},\hat{\boldsymbol{\beta}},\hat{\boldsymbol{\theta}},\hat{\mathbf{x}}\right) \in \hat{\mathrm{A}}\hat{\mathrm{B}}\hat{\Theta}\hat{X}$ obtained from the subproblem in all iterations up to the current one, we can solve the lower-bounding master problem:

[$DAD^{dLP}m\hat{x}\hat{x}$-CN-MP]

$$\underline{z}(\hat{\mathrm{A}}\hat{\mathrm{B}}\hat{\Theta}\hat{X}) = \min_{\mathbf{w}\in W} z \tag{77}$$

$$\text{s.t. } z \ge (\mathbf{b}^y)^T\hat{\boldsymbol{\alpha}} + (\mathbf{1}-\hat{\mathbf{x}}+\mathbf{w})^T\hat{\boldsymbol{\beta}} + \mathbf{u}^T\hat{\boldsymbol{\theta}} \ \ \forall (\hat{\boldsymbol{\alpha}},\hat{\boldsymbol{\beta}},\hat{\boldsymbol{\theta}},\hat{\mathbf{x}}) \in \hat{\mathrm{A}}\hat{\mathrm{B}}\hat{\Theta}\hat{X} \tag{78}$$

where the set $\hat{\mathrm{A}}\hat{\mathrm{B}}\hat{\Theta}\hat{X} \in \mathrm{AB}\Theta X$ enumerates all outcomes identified by the subproblem on every iteration.

1. An Algorithm to Solve *DAD*-CN

The following algorithm solves the reformulation-based *DAD* model.

Algorithm 2:

Input: An instance of [*DADmxm*-CN1], an allowable optimality gap ε, and any feasible defense plan $\hat{\mathbf{w}}^0$ (e.g., $\hat{\mathbf{w}} = \mathbf{0}$).

Output: An ε-optimal defensive plan $\mathbf{w}^*$ for [*DADmxm*-CN1], as well as the optimal attack plan $\mathbf{x}^*$ and the optimal system-operation plan $\mathbf{y}^*$.

{

Initialize: $\overline{z} \leftarrow \infty$; $\underline{z} \leftarrow -\infty$; $\hat{\mathbf{w}} \leftarrow \hat{\mathbf{w}}^0$; $\hat{\mathrm{A}}\hat{\mathrm{B}}\hat{\Theta}\hat{X} \leftarrow \varnothing$

While $\left(\overline{z} - \underline{z} > \varepsilon\right)$ {

Solve [$DAD^{dLP}\hat{m}xx$-CN-SP] with input $\hat{\mathbf{w}}$ (see equations (73)-(76)) to obtain an incumbent upper bound on the objective function $\overline{z}(\hat{\mathbf{w}})$, the attack plan $\hat{\mathbf{x}}$ and the dual variables $\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\theta}}$;

$\hat{\mathrm{A}}\hat{\mathrm{B}}\hat{\Theta}\hat{X} \leftarrow \hat{\mathrm{A}}\hat{\mathrm{B}}\hat{\Theta}\hat{X} \cup \left\{\left(\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\theta}}, \hat{\mathbf{x}}\right)\right\}$;

If $\left(\overline{z}(\hat{\mathbf{w}}) < \overline{z}\right)$ {

$\overline{z} \leftarrow \overline{z}(\hat{\mathbf{w}})$; $\mathbf{w}^* \leftarrow \hat{\mathbf{w}}$;

If $\left(\overline{z} - \underline{z} \le \varepsilon\right)$ break from While loop;

}

Solve [$DAD^{dLP}m\hat{x}\hat{x}$-CN-MP] with input $\left(\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\theta}}, \hat{\mathbf{x}}\right) \in \hat{\mathrm{A}}\hat{\mathrm{B}}\hat{\Theta}\hat{X}$ (See equations (77)-(78)), to obtain $\underline{z}(\hat{\mathrm{A}}\hat{\mathrm{B}}\hat{\Theta}\hat{X})$ and a new defense plan $\hat{\mathbf{w}}$;

$\underline{z} \leftarrow \underline{z}(\hat{\mathrm{A}}\hat{\mathrm{B}}\hat{\Theta}\hat{X})$;

}

Print ("ε-optimal defense plan, activity levels and objective-function values are" $\mathbf{w}^*$, $\mathbf{y}^*$, z^*, "respectively.");

}

2. Implementation of DAD-CN Model for DSP

We now proceed to implement *DAD* in its capacity-interdiction version for the *DSP*. We do not allow the defender-operator to traverse interdicted arcs, so we put into effect the first of the two models $DAD^{LP}mxm$-CN1.

Indices and index sets. The same as those used in [$DAD^{LP}mxm$]

Data. We must include here:

$u_k = 1$ Nominal capacity of arc (i, j).

Variables:

α_i Dual variable for the operative constraints on each node i.

β_k Dual variables for capacity-interdiction constraints on every arc k

β'_k Auxiliary variables for capacity-interdiction constraints

θ_k Dual variables for the max-flow constraints on every arc

Formulation of the basic problem for DSP:

[$DAD^{dLP}mxx$-CN]

$$\min_{\mathbf{w}\in\mathbf{W}_{DSP}} \max_{\substack{\mathbf{x}\in X_{DSP} \\ \boldsymbol{\beta}\le\mathbf{0},\boldsymbol{\theta}\le\mathbf{0}}} \quad \alpha_s - \alpha_t + \sum_{k\in\mathcal{A}} \left(1 - x_k + w_k\right)\beta_k + \sum_{k\in\mathcal{A}} \theta_k u_k$$

$$\text{s.t.} \quad \alpha_j - \alpha_i + \beta_k + \theta_k \le c_k \quad \forall\, k = (i,j) \in \mathcal{A}$$

Formulation of the subproblem for DSP:

[$DAD^{dLP}\hat{m}xx$-CN-SP]

$$\bar{z}(\hat{\mathbf{w}}) = \max_{\mathbf{x},\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\theta}} \quad \alpha_s - \alpha_t + \sum_{k|\hat{w}_{i,j}=1} 2\beta_k + \sum_{k|\hat{w}_k=0} \beta'_{i,j} + \sum_{k\in\mathcal{A}} \theta_k u_k$$

$$\text{s.t.} \quad \alpha_j - \alpha_i + \beta_k + \theta_k \le c_k \quad \forall\, k = (i,j) \in \mathcal{A} \tag{79}$$

$$\beta'_k \ge -M\left(1 - x_k\right) \qquad \forall\, k \in \mathcal{A} \mid \hat{w}_k = 0 \tag{80}$$

$$\beta'_k \le \beta_k + Mx_k \qquad \forall\, k \in \mathcal{A} \mid \hat{w}_k = 0 \tag{81}$$

$$\mathbf{x} \in X_{DSP} \tag{82}$$

$$\beta_k \le 0,\ \beta'_k \le 0,\ \theta_k \le 0,\ \alpha_k \text{ free}$$

Constraints (79) are the optimality conditions for DSP; constraints (80) and (81) are used to linearize the model; and constraint (82) is the attacker's resource constraint.

Formulation of the master problem for DSP:

$[DAD^{dLP}m\hat{x}\hat{x}\text{-CN-MP}]$

$$\underline{z}(\hat{\mathrm{A}}\hat{\mathrm{B}}\hat{\Theta}\hat{X}) = \min_{\mathbf{w},z} z$$

$$\text{s.t. } z \geq \hat{\alpha}_s - \hat{\alpha}_t + \sum_{k\in\mathcal{A}} (1-\hat{x}_k + w_k)\hat{\beta}_k + \sum_{k\in\mathcal{A}} \hat{\theta}_k u_k \quad \forall \left(\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\theta}}, \hat{\mathbf{x}}\right) \in \hat{\mathrm{A}}\hat{\mathrm{B}}\hat{\Theta}\hat{X}$$

$$\mathbf{w} \in W_{DSP}$$

C. THE ATTACKER-DEFENDER-DEFENDER MODEL.

This section proposes a method to solve the Attacker-Defender-Defender model in the capacity-interdiction version sketched in Chapter III. The purpose is to obtain a lower bound on z^* for *DAD*.

When we outlined the model, we mentioned that we need to use bounds to solve bi-level MIPs. For that reason, a decomposition algorithm seems appropriate. Since the outer layer is a maximization problem, any feasible $\hat{\mathbf{x}}$ leads to a lower bound on z^*. This bound can be calculated by solving the following subproblem:

$[ADD^{LP}\hat{x}mm\text{-SP}]$ $\quad \underline{z}(\hat{\mathbf{x}}) = \min_{\mathbf{w}\in W,\, \mathbf{y}\geq\mathbf{0},\, \mathbf{y}'\geq\mathbf{0}} \mathbf{cy} + (\mathbf{c}+\mathbf{d})\mathbf{y}'$ (83)

$$\text{s.t.} \quad A^y\mathbf{y} + A^y\mathbf{y}' = \mathbf{b}^y \qquad [\boldsymbol{\alpha}] \qquad (84)$$

$$\mathbf{y} \leq U(\mathbf{1} - \hat{\mathbf{x}} + \mathbf{w}) \qquad [\boldsymbol{\beta}] \qquad (85)$$

$$\mathbf{y} + \mathbf{y}' \leq \mathbf{u} \qquad [\boldsymbol{\theta}] \qquad (86)$$

As stated in the previous chapter, the inclusion of the second set of constraints allows us to model an interdicted activity *k* that cannot be used if it has been attacked $(\hat{x}_k = 1)$, unless it is defended. (Specifically for DSP, $(y_k = 1) \Rightarrow (w_k = 1)$). However, since the activity is artificially doubled by y'_k, it is still possible to use it, but only if the corresponding per-unit penalty d_k is paid. This alternative action is characterized by setting the variable $y'_k = 1$. The third constraint keeps the values of $\mathbf{y}$ and $\mathbf{y}'$ within the capacity limits. (For certain type of problems such as DSP, we can assume that, since $\mathbf{w}$ is binary, $\mathbf{y} \geq \mathbf{0}$ and $\mathbf{y}' \geq \mathbf{0}$ will lead to binary solutions.)

The vectors $\hat{\mathbf{y}}$ and $\hat{\mathbf{w}}$ extracted from the solution to the subproblem (we need to form a new $\hat{\mathbf{y}} \leftarrow \hat{\mathbf{y}} + \hat{\mathbf{y}}'$), are now useful to compute an upper bound, which will be given by the optimum value of the following master problem:

$$[ADD^{LP}x\hat{m}\hat{m}\text{-MP}]\ \overline{z}(\hat{W}\hat{Y}) = \max_{\mathbf{x}\in X} z \tag{87}$$

$$\text{s.t. } z \le \left(\mathbf{c}^T + \left(\mathbf{x}^T - \hat{\mathbf{w}}^T\right)^+ D\right)\hat{\mathbf{y}} \quad \forall (\hat{\mathbf{w}}, \hat{\mathbf{y}}) \in \hat{W}\hat{Y} \tag{88}$$

Here, constraints (88) represents Benders cuts based on the pair $(\hat{\mathbf{w}}, \hat{\mathbf{y}}) \in \hat{W}\hat{Y}$ coming from all previous SP solutions. The set $\hat{W}\hat{Y}$ denotes the $(\hat{\mathbf{w}}, \hat{\mathbf{y}})$ pairs identified by the algorithm. When $\hat{W}\hat{Y} \subset WY$, the master problem is a relaxation of $ADD^{LP}xmm$ and $\mathbf{z}(\hat{\mathbf{w}}, \hat{\mathbf{y}})$, which denotes the solution of the master problem given $(\hat{\mathbf{w}}, \hat{\mathbf{y}})$, is an upper bound on the objective value.

1. An Algorithm to Solve *ADD*

In the following two sub-sections, we propose an algorithm to solve the reordering-based *ADD* using decomposition, and implement the algorithm for *DSP*.

Algorithm 3:

Input: An instance of *ADDxmm* and an allowable optimality gap ε, any feasible attack plan $\hat{\mathbf{x}}^0$ (e.g., $\hat{\mathbf{x}}^0 = \mathbf{0}$).

Output: An ε-optimal defensive plan $\mathbf{w}^*$ for $ADD^{LP}xmm$, as well as the optimal attack plan $\mathbf{x}^*$, optimal system-operation plan $\mathbf{y}^*$ and a lower bound for *DAD* ($\underline{\hat{z}}_{DAD} \leftarrow z^*_{ADD}$).

{

Initialize: $\overline{z} \leftarrow \infty$; $\underline{z} \leftarrow -\infty$; $\hat{\mathbf{x}} \leftarrow \hat{\mathbf{x}}^0$; $\hat{W}\hat{Y} \leftarrow \varnothing$;

While $\left(\overline{z} - \underline{z} > \varepsilon\right)$ {

Solve $[ADD^{LP}\hat{x}mm\text{-SP}]$ with input $\hat{\mathbf{x}}$ to obtain an incumbent lower bound $\underline{z}(\hat{\mathbf{x}})$, defense plan $\hat{\mathbf{w}}$, and operating plan $\hat{\mathbf{y}} \leftarrow \hat{\mathbf{y}} + \hat{\mathbf{y}}'$;

$\hat{W}\hat{Y} \leftarrow \hat{W}\hat{Y} \cup \{(\hat{\mathbf{w}}, \hat{\mathbf{y}})\}$;

If $(\underline{z}(\hat{\mathbf{x}}) > \underline{z})$ {

$\underline{z} \leftarrow \underline{z}(\hat{\mathbf{x}})$; $\mathbf{x}^* \leftarrow \hat{\mathbf{x}}$;

If $(\overline{z} - \underline{z} \le \varepsilon)$ break from While loop;

}

Solve [$ADD^{LP}x\hat{m}\hat{m}$-MP] for all$(\hat{\mathbf{w}}, \hat{\mathbf{y}})$ derived from $\hat{W}\hat{Y}$, to obtain an upper bound $\overline{z}(\hat{W}\hat{Y})$ and a new attack plan $\hat{\mathbf{x}}$;

$\overline{z} \leftarrow \overline{z}(\hat{W}\hat{Y})$;

}

Print ("*ADD* ε-optimal defense plan, activity levels and *DAD* lower bound values are" $\mathbf{w}^*$, $\mathbf{y}^*$, z^*, "respectively.");

}

2. Implementation of *ADD* for DSP

Problem definition. The Shortest Path Problem (DSP), as defined in Section A.

Indices and index sets. The same as those used in [$DAD^{LP}mxm$]

Data. The new data with respect to [$DAD^{LP}mxm$] are:

$u_k = 1$ Nominal capacity of arc k

Variables. Here, we must add:

y'_k Defender's decision to traverse arc k which has been attacked and not protected. $y'_k = 0$ if arc k is traversed, and $y'_k = 0$ otherwise.

Formulation of the basic problem for DSP: As in the previous subsections and, for clarity, we start with the formulation of the basic problem:

$$[ADD^{LP}xmm] \quad \max_{\mathbf{x}\in X_{DSP}} \min_{\substack{\mathbf{w}\in W_{DSP}\\ \mathbf{y},\mathbf{y}'}} \sum_k c_k y_k + (c_k + d_k) y'_k$$

$$\text{s.t.} \sum_{k\in\mathcal{FS}(i)} (y_k + y'_k) - \sum_{k\in\mathcal{RS}(i)} (y_k + y'_k) = \begin{cases} 1 & \text{if } i = s \\ 0 & \forall i \in \mathcal{N}\backslash\{\text{s,t}\} \\ -1 & \text{if } i = t \end{cases}$$

$$y_k \le (1 - x_k + w_k) \;\; \forall k \in \mathcal{A}$$

$$y_k + y'_k \le 1 \qquad \forall k \in \mathcal{A}$$

Formulation of the subproblem for DSP:

$[ADD^{LP}\hat{x}mm\text{-SP}]$

$$\overline{z}(\hat{\mathbf{x}}) = \min_{\substack{\mathbf{w}\in W_{DSP}\\ \mathbf{y},\mathbf{y}'}} \sum_{k\in\mathcal{A}} c_k y_k + (c_k + d_k) y'_k \tag{89}$$

$$\text{s.t.} \quad \sum_{k\in\mathcal{FS}(i)} (y_k + y'_k) - \sum_{k\in\mathcal{RS}(i)} (y_k + y'_k) = \begin{cases} 1 & \text{if } i = s \\ 0 & \forall i \in \mathcal{N}\backslash\{\text{s,t}\} \\ -1 & \text{if } i = t \end{cases} \tag{90}$$

$$y_k \le (1 - \hat{x}_k + w_k) \qquad \forall k \in \mathcal{A} \tag{91}$$

$$y_k + y'_k \le 1 \qquad \forall k \in \mathcal{A} \tag{92}$$

$$y_k \ge 0,\; y'_k \ge 0 \qquad \forall k \in \mathcal{A} \tag{93}$$

Constraints (90) represent standard flow-balance constraints for a shortest-path problem. Constraints (91) are the capacity-interdiction constraints for every arc (although, for fixed $\hat{\mathbf{x}}$, these are actually capacity-expansion constraints). Constraints (92) are the flow capacity constraints. Since $u_{i,j} = 1$ (already implemented in the model), either $y_{i,j}$ or $y'_{i,j}$ must be chosen by the defender-operator.

Formulation of the master problem for DSP:

$[ADD^{LP}x\hat{m}\hat{m}\text{-MP}]$

$$\overline{z}(\hat{W}\hat{Y}) = \max_{\mathbf{x}\in X_{DSP},z} \; z \tag{94}$$

$$\text{s.t.} \quad z \le \sum_{k\in\mathcal{A}} \left(c_k + x_k(1 - \hat{w}_k)d_k\right)\hat{y}_k \;\; \forall \left(\hat{\mathbf{w}},\hat{\mathbf{y}}\right) \in \hat{W}\hat{Y} \tag{95}$$

D. A SPECIALIZED ALGORITHM TO SOLVE *DSP*

The previous section illustrated the implementation of *ADD* for *DSP*. Let *DSPxmm* be an instance of [*ADDxmm*] where attacker and defender have cardinality constraints on their actions. Then [*DSPxmm*] may be viewed as follows:

- The attacker finds an interdiction plan $\hat{\mathbf{x}}$;
- The defender chooses up to b^w interdicted arcs, whose cost is $(c_k + d_k)$, and converts them back to their original cost c_k; and
- Finally, the defender-operator solves the shortest-path problem through the network.

A max-min-min is just a max-min where the inner two stages (the defense and the system operation) are carried out simultaneously. The problem can be envisioned as a type of network-interdiction problem and solved as the "Maximizing the Shortest-path" ("*MXSP*"; see Israeli and Wood 2002) in an expanded network with the following structure:

The network $\mathcal{G}$ of Figure 4 is expanded in levels as shown in Figure 5. Essentially, each level is a copy of the original network. In addition, an extra set of arcs, denoted by $k' \in \mathcal{A}_1^+$, links levels and allows the defender to jump from one level to the next. These "between-level" arcs k' are not subject to interdiction and each mimics its fellow "same-level" arc. However, each head points to the corresponding node in the next higher level. The number of levels equals the number of defensive resource available plus one. (For example, if the defender has four units of defense resources, the network gets expanded in levels zero through four, i.e., $\mathcal{L} = \{0,1,2,...,\overline{L}\}$ where $\overline{L} = b^w = 4$). In this particular example, the defender has one unit of resource $(b^w = 1)$ and the network is expanded by only one level.

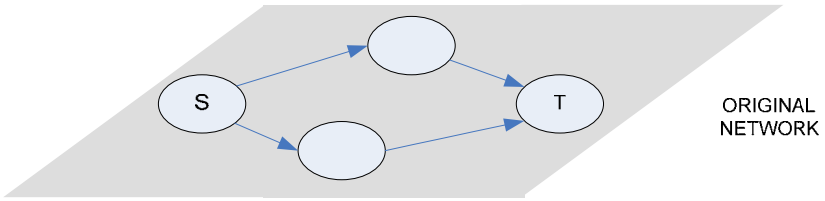

Figure 4. Original network $\mathcal{G}$ with 4 nodes and 4 arcs that represents a hypothetical shortest-path problem that the defender must solve.

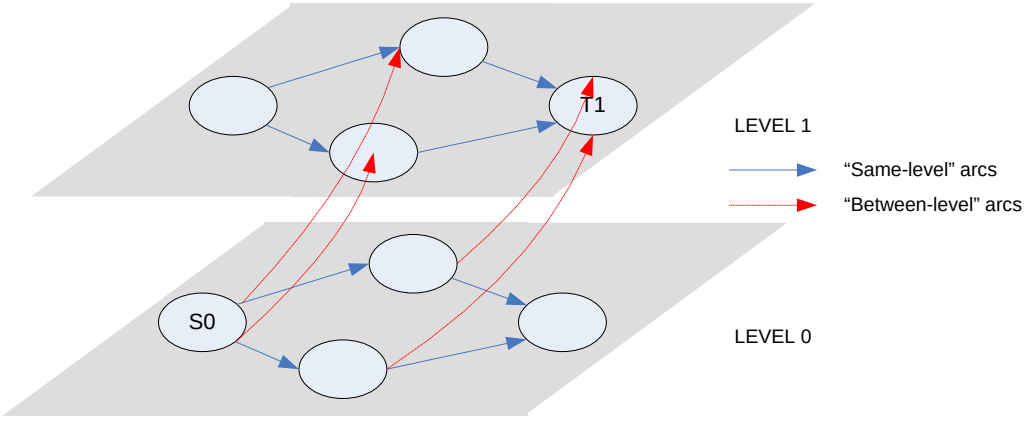

Figure 5. Expanded network $\mathcal{G}^+$ when $b^w = 1$. Solving the network-interdiction problem in $\mathcal{G}^+$ is equivalent to solving *ADD* in $\mathcal{G}$.

If $\mathcal{N}$ denotes the set of all nodes in the original network, $\mathcal{N}^+ = \mathcal{N} \times \mathcal{L}$ is the set of all nodes in $\mathcal{G}^+ = (\mathcal{N}^+, \mathcal{A}^+)$, that is, $\mathcal{N}^+ = \bigcup_{i \in \mathcal{N}, l \in \mathcal{L}} \{i_0, i_1, ..., i_{\bar{L}}\}$.

Likewise, if $\mathcal{A}$ represents the set of all arcs k in the original network, $\mathcal{A}_0^+$ is the set of all "same-level" arcs in $\mathcal{G}^+$, $\mathcal{A}_0^+ = \bigcup_{k \in \mathcal{A}} \{k_0, k_1, ..., k_{\bar{L}})\}$; and $\mathcal{A}_1^+$ is the set of all between-level arcs, $\mathcal{A}_1^+ = \bigcup_{k' \in \mathcal{A}^+} \{k'_0, k'_1, ..., k'_{\bar{L}-1}\}$. Then, $\mathcal{A}^+ = \mathcal{A}_0^+ \cup \mathcal{A}_1^+$.

The defender-operator must traverse from the source node at level zero to the sink node located in the uppermost level.

$$[MXSP^{LP}] \qquad \max_{\mathbf{x} \in X_{DSP}} \min_{\mathbf{y} \geq \mathbf{0}, \mathbf{y}' \geq \mathbf{0}} \quad (\mathbf{c}^T + \mathbf{x}^T D)\mathbf{y} + \mathbf{c}^T \mathbf{y}' \tag{96}$$

$$\text{s.t.} \qquad A^y \mathbf{y} + A^{y'} \mathbf{y}' = \mathbf{b}_l^y \ \ \forall l \in \mathcal{L} \qquad [\boldsymbol{\pi}_l] \tag{97}$$

The new variable $\mathbf{y}'$ corresponds to the defender-operator's decision to jump from one level to the next, skipping any possible interdiction and paying the original cost. The functioning of the variable $\mathbf{y}'$ is very similar to the $\mathbf{y}'$ used in *ADD* (see equations (83)-(84)). However, in this case, we do not need a constraint to control the expenditure of defense resource. This is because the structure of the new network itself will force the defender to pick exactly b^w arcs to defend.

Observe now that, as opposed to the generic models, after fixing $\mathbf{x}$, everything is linear in the objective function. If we take the dual of the inner min problem, we obtain the following MIP:

$$[MXSP^{dLP}] \quad \max_{\mathbf{x}\in X_{DSP}} \max_{\boldsymbol{\pi}} \quad (\mathbf{b}_l^y)^T \boldsymbol{\pi}_l \qquad (98)$$

$$\text{s.t.} \quad (A^y)^T \boldsymbol{\pi}_l \le \mathbf{c} + D\mathbf{x} \qquad [\mathbf{y}] \qquad (99)$$

$$(A^{y'})^T \boldsymbol{\pi}_l \le \mathbf{c} \qquad [\mathbf{y}'] \qquad (100)$$

$$\boldsymbol{\pi} \text{ free}$$

Since this is still an NP-hard MIP (Israeli and Wood 2002), it may be difficult to solve for large problems. Because we are expanding the network by adding more levels according to the number of defensive resources, the number of decision variables increases. A simple, square-lattice network with 25 nodes on each side has only 2,400 arcs, but if $b^w = 10$, $[MXSP^{dLP}]$ has 45,600 variables.

1. Implementation of MXSP in $\mathcal{G}^+$

Problem definition. Maximize the shortest *s*-*t* path in an expanded directed network $\mathcal{G}^+$ by interdicting arcs. (Note that the formulation uses data and notation from the original network $\mathcal{G}$ rather than the expanded network $\mathcal{G}^+$).

Indices and index sets:

$i, j \in \mathcal{N}$ Nodes in $\mathcal{G} = (\mathcal{N}, \mathcal{A})$

$l \in \mathcal{L}$ Levels $\mathcal{L} = \{0,1,...,\overline{L}\}$ where $\overline{L} = b^w$ (defense resources)

$k \in \mathcal{A}$ Arcs in $\mathcal{G} = (\mathcal{N}, \mathcal{A})$

s	Source node in $\mathcal{G}=(\mathcal{N},\mathcal{A})$, located at level 0
t	Sink node in $\mathcal{G}=(\mathcal{N},\mathcal{A})$, located at level $\bar{L}$
$\mathcal{FS}(i)$	Forward Star of node $i \in \mathcal{N}$
$\mathcal{RS}(i)$	Reverse Star of node $i \in \mathcal{N}$

Data: Similar as those used in the *DAD* implementation (Section 4)

Variables:

$y_{k,l}$ Defender's decision to traverse arc k at level l. $y_{k,l}=1$ if arc is traversed, $y_{k,l}=0$ otherwise

$y'_{k,l}$ Defender's decision to defend and traverse arc k between levels l and $l+1$. $y'_{k,l}=1$ if arc is traversed, $y'_{k,l}=0$ otherwise

Problem Formulation:

$[MXSP^{LP}]$

$$\max_{\mathbf{x}\in X_{DSP}} \min_{\mathbf{y},\mathbf{y}'} \sum_{k\in\mathcal{A}}\sum_{l\in\mathcal{L}}\left(c_k + d_k x_k\right) y_{k,l} + \sum_{k\in\mathcal{A}}\sum_{l\in\mathcal{L}\setminus\{\bar{L}\}} c_k y'_{k,l} \tag{101}$$

$$\text{s.t.} \sum_{k\in\mathcal{FS}(i)} (y_{k,l}+y'_{k,l}) - \sum_{k\in\mathcal{RS}(i)} (y_{k,l}+y'_{k,l}) = \begin{cases} 1 & \text{for } i=s \text{ and } l=0 \\ 0 & \forall\,(i,l)\in\mathcal{N}\times\mathcal{L}\setminus\{(s,0),(t,\bar{L})\} \\ -1 & \text{for } i=t \text{ and } l=\bar{L} \end{cases} \tag{102}$$

$$y_{k,l} \geq 0 \;\forall\, k\in\mathcal{A},\ \forall\, l\in\mathcal{L} \tag{103}$$

$$y'_{k,l} \geq 0 \;\forall\, k\in\mathcal{A},\ \forall\, l\in\mathcal{L}\setminus\{\bar{L}\} \tag{104}$$

Constraints (102) are the flow-balance constraints on every node in $\mathcal{G}^+$.

Formulation of dual problem:

$[MXSP^{dLP}]$

$$\max_{\mathbf{x},\boldsymbol{\pi}} \quad \pi_{t,\bar{L}} - \pi_{s,0} \tag{105}$$

$$\text{s.t.} \quad \pi_{i,l} - \pi_{j,l} - d_k x_k \le c_k \quad \forall\, k=(i,j) \in \mathcal{A},\ \forall\, l \in \mathcal{L} \qquad [y_{k,l}] \tag{106}$$

$$\pi_{i,l} - \pi_{j,l+1} \le c_k \quad \forall\, k=(i,j) \in \mathcal{A},\ \forall\, l \in \mathcal{L} \setminus \{\bar{L}\} \qquad [y'_{k,l}] \tag{107}$$

$$\pi_{s,0} = 0 \tag{108}$$

$$\mathbf{x} \in X_{DSP} \tag{109}$$

$$\boldsymbol{\pi} \text{ free}$$

Constraints (106) and (107) are optimality constraints for DSP, and constraint (108) normalizes the dual variables (this is valid since the inner min problem has one redundant flow balance constraint).

Chapter V shows computational results obtained for a batch of hypothetical networks of different size and shape. It, also, investigates the quality of the bound it produces with respect to *DAD*. Further, it shows the differences from the more generally applicable *ADD* solution method implemented in Section C.

2. Solving *MXSP* by Decomposition [$MXSP^{LP}$-D]

Another possible approach to solving *MXSP* is using a decomposition method. In doing so, we will be dealing with constrained shortest paths on one hand, and the typical cuts of a master problem on the other. Since that outer layer is a maximization problem, any feasible attack plan ($\hat{\mathbf{x}}$) will give us a lower bound on the objective function. We can get this bound by solving the following subproblem:

$$[MXSP^{LP}\text{-SP}] \quad \underline{z}(\hat{\mathbf{x}}) = \min_{\mathbf{y}\ge\mathbf{0},\ \mathbf{y}'\ge\mathbf{0}} \left(\mathbf{c}^T + \hat{\mathbf{x}}^T D\right)\mathbf{y} + \mathbf{c}^T\mathbf{y}' \tag{110}$$

$$\text{s.t.} \quad A^y\mathbf{y} + A^{y'}\mathbf{y} = \mathbf{b}_l \ \ \forall\, l \in L \qquad [\boldsymbol{\pi}_l] \tag{111}$$

Let $Y'Y = Y' \times Y$ denote the set that contains all possible pairs of $(\mathbf{y}', \mathbf{y})$. In addition, $\hat{Y}'\hat{Y} \subseteq Y'Y$ is just a subset of $Y'Y$, where only certain defensive/utilization pairs have been identified by the following master problem:

$$[MXSP^{LP}\text{-}MP] \quad \overline{z}(\hat{Y}'\hat{Y}) = \max_{\mathbf{z},\mathbf{x}\in X_{DSP}} \quad z \tag{112}$$

$$\text{s.t.} \quad z \le +\left(\mathbf{c}^T + \mathbf{x}^T D\right)\hat{\mathbf{y}} + \mathbf{c}^T\hat{\mathbf{y}}' \quad \forall(\hat{\mathbf{y}}',\hat{\mathbf{y}}) \in \hat{Y}'\hat{Y} \tag{113}$$

The solution of this master problem, as with previously discussed decomposition algorithms (see Section IV.C), yields an upper bound on z^*.

The subproblem is an LP with an totally unimodular constraint matrix (Ahuja, Magnanti, and Orlin, 1993, pp. 447-449). Thus, the variables $\mathbf{y}$ and $\mathbf{y}'$ only need to be non-negative and continuous, and will adopt a binary values intrinsically.

However, the master problem is still a MIP and the vector $\mathbf{x}$ (the attacker's decision variables) must be binary. Essentially, we are dealing with another difficult problem. We might try to tighten it by using of well-known techniques to reduce the size of the feasible region for the LP relaxation of the MIP, adding integer cutting planes, but that is beyond the scope of this thesis.

a. An Algorithm to Solve MXSP by Decomposition

The next two subsections propose a decomposition algorithm to solve $[MXSP^{LP}\text{-D}]$.

Algorithm 4:

Input: An instance of $[ADDxmm]$ ($[DSPxmm]$), preprocessed by expanding the network and transformed into an instance of *MXSP*, and any feasible attack plan $\hat{\mathbf{x}}^o$ (e.g., $\hat{\mathbf{x}}^o = \mathbf{0}$).

Output: An ε-optimal defensive plan $\mathbf{y}'^*$ for *ADD*, the optimal attack plan $\mathbf{x}^*$, the optimal way to operate the system following a worst-case attack $\mathbf{y}^*$, and a lower bound for *DAD*, $\underline{\hat{z}}_{DAD} \leftarrow z^*_{MXSP}$.

{

Initialize: $\overline{z} \leftarrow \infty$; $\underline{z} \leftarrow -\infty$; $\hat{\mathbf{x}} \leftarrow \hat{\mathbf{x}}^0$; $\hat{Y}'\hat{Y} \leftarrow \varnothing$;

While $\left(\overline{z} - \underline{z} > \varepsilon\right)$ {

Solve [$MXSP^{LP}$-SP] with input $\hat{\mathbf{x}}$ to obtain $\underline{z}(\hat{\mathbf{x}})$, and the optimal combination of defensive and operating plans $\hat{\mathbf{y}}', \hat{\mathbf{y}}$; $\hat{Y}'\hat{Y} \leftarrow \hat{Y}'\hat{Y} \cup \{(\hat{\mathbf{y}}', \hat{\mathbf{y}})\}$;

If $\left(\underline{z}(\hat{\mathbf{x}}) > \underline{z}\right)\{$

$\underline{z} \leftarrow \underline{z}(\hat{\mathbf{x}})$; $\mathbf{x}^* \leftarrow \hat{\mathbf{x}}$;

If $\left(\overline{z} - \underline{z} \le \delta\right)$ break from While loop;

}

Solve [$MXSP^{LP}$-MP] for all $(\hat{\mathbf{y}}', \hat{\mathbf{y}}) \in \hat{Y}'\hat{Y}$, to obtain $\overline{z}(\hat{Y}'\hat{Y})$ and the next attack plan $\hat{\mathbf{x}}$;

$\overline{z} \leftarrow \overline{z}(\hat{Y}'\hat{Y})$;

}

Print ("ε-optimal defense plan for *ADD*, activity levels and lower bound for *DAD* are" $\mathbf{y}'^*$, $\mathbf{y}^*$, z^*, "respectively.");

}

b. Implementation of MXSP in $\mathcal{G}^+$ by Decomposition

Indices, sets, parameters, and variables of this formulation are the same as those proposed in the direct implementation of *MXSP*. The first subproblem is implemented as follows:

[$MXSP^{LP}$-SP]

$$\underline{z} = \min_{\mathbf{y},\mathbf{y}'} \sum_{k\in\mathcal{A}} \sum_{l\in\mathcal{L}} \left(c_k + d_k \hat{x}_k\right) y_{k,l} + \sum_{k\in\mathcal{A}} \sum_{l\in\mathcal{L}\setminus\{\bar{L}\}} c_k y'_{k,l} \tag{114}$$

$$\text{s.t.} \sum_{k\in\mathcal{FS}(i)} (y_{k,l} + y'_{k,l}) - \sum_{k\in\mathcal{RS}(i)} (y_{k,l} + y'_{k,l}) = \begin{cases} 1 \text{ for } i = s, l = 0 \\ 0 \ \forall (i,l) \in \mathcal{N}\times\mathcal{L}\setminus\{(s,0),(t,\bar{L})\} \\ -1 \text{ for } i = t, l = \bar{L} \end{cases}$$

$$y_{k,l} \ge 0 \ \forall k \in \mathcal{A},\ l \in \mathcal{L}$$

$$y'_{k,l} \geq 0 \; \forall \, k \in \mathcal{A}, \; l \in \mathcal{L} \backslash \{\bar{L}\}$$

The master problem is:

$$[MXSP^{LP}\text{-MP}]$$

$$\bar{z}(\hat{\mathbf{y}}', \hat{\mathbf{y}}) = \max_{\mathbf{x} \in X_{DSP}} z \tag{115}$$

$$\text{s.t.} \quad z \leq \sum_{k \in \mathcal{A}} \sum_{l \in \mathcal{L}} \left(c_k + d_k x_k \right) \hat{y}_{k,l} + \sum_{k \in \mathcal{A}} \sum_{l \in \mathcal{L} \backslash \{\bar{L}\}} c_k \hat{y}'_{k,l} \quad \forall \, (\hat{\mathbf{y}}', \hat{\mathbf{y}}) \in \hat{Y}' \hat{Y} \tag{116}$$

THIS PAGE INTENTIONALLY LEFT BLANK

V. COMPUTATIONAL RESULTS

This chapter presents results obtained by testing the four algorithms described above against hypothetical grid networks, with fixed structure and randomly generated arc attributes. The grids are created using Java code (Sun Microsystems 2004) using the pseudo-random number generator included in Java's class "Random." The algorithms are implemented in GAMS (GAMS Development Corporation 2007).

A. COMPUTATIONAL RESULTS FOR *DAD* MODELS

This section shows the results when *DAD* is implemented to solve DSP (Defending the Shortest Path, i.e., the tri-level shortest-path defense problem) for a network with the following characteristics:

- Square topology similar as Figure 6 with a 10×10 grid of nodes (plus source and sink).
- There is one source node *s* and one sink node *t*. Arcs k departing and arriving to these nodes have cost $c_k = 1$ and interdiction penalty $d_k = 0$.

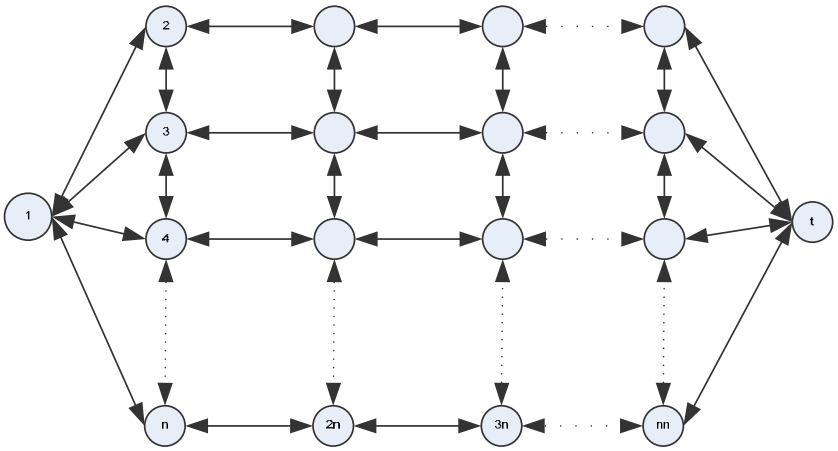

Figure 6. Network square topology with an *n*×*n* grid of nodes.

- Other arc costs c_k and penalties d_k and penalties, except those mentioned above, are randomly, uniformly distributed on [0,1] and [1,2] respectively.

The XPRESS solver is used within GAMS (XPRESS Solver Manual, GAMS 2007), with an absolute termination criterion of $\varepsilon_1 = 0.01$, i.e., 1%, and the same value for the allowable decomposition gap ($\varepsilon_2 = 0.01$). (Tests for $\overline{z} - \underline{z} \le \varepsilon$ in the algorithms are replaced by $(\overline{z} + \varepsilon_1) - (\underline{z} - \varepsilon_1) \le \underline{z}\varepsilon_2$.

The first experiment solves DSP as an instance of $[DAD^{LP}]$ using Algorithm 1 and 1A, with the subproblem solved directly (standard decomposition) and by nested decomposition respectively.

Table 1 provides summary statistics of execution time in seconds, elapsed time (which includes equation-generating time and other overhead), and number of iterations, for different interdiction and defensive integer resources. Appendix II displays a complete table with all combinations of attack and defensive resource ranging from two to seven.

		Standard decomposition Algorithm 1			**Nested decomposition Algorithm 1A**			**Ratio**
Attack	Defen.	CPU time A (sec.)	Elapsed time (sec.)	Iter.	CPU time B (sec.)	Elapsed time sec.)	Iter.	A/B
2	2	2.7	4.2	4	1.2	7.5	4	2.30
3	3	21.5	24.8	8	10.7	31.0	7	2.01
4	4	109.6	114.8	11	57.3	124.9	14	1.91
5	5	396.6	406.4	21	306.0	484.2	28	1.30
6	6	3047.0	3067.6	43	1719.0	2075.9	32	1.77
7	7	6278.9	6298.9	54	1490.0	1674.6	16	4.21
							Avg:	**1.87**

Table 1. Computational results for DSP using Algorithm 1 and 1A (Section IV.A.3.). The "standard decomposition" solves each subproblem directly, i.e., using LP-based branch-and-bound on the subproblem MIP. "Nested decomposition" solves the subproblem by Benders decomposition. "Ratio" represents the improvement in CPU times for the nested decomposition method with respect to the direct decomposition one.

On average, the nested decomposition runs 1.87 times faster than the standard decomposition. (We compute this improvement with respect to CPU time, because most of the extra elapsed time could be recovered by a more efficient implementation that avoided the GAMS overhead).

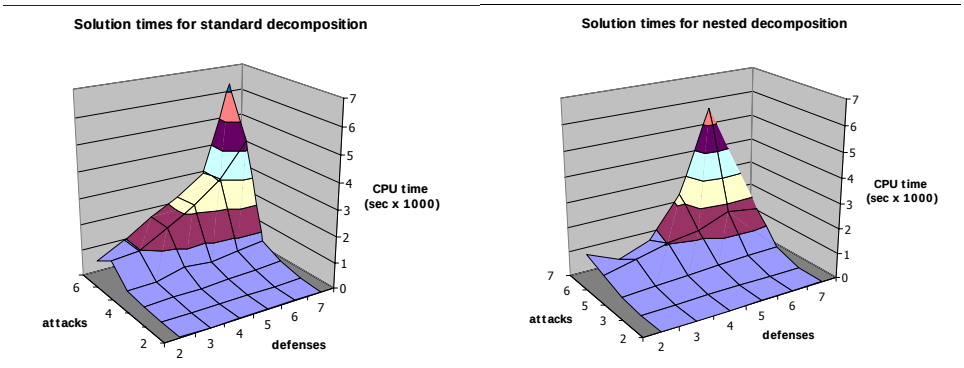

Figure 7. CPU times for DSP problem using Algorithm 1. With less volume under the surface, which indicates fewer seconds to execute, the nested decomposition proves to outperform the standard *DAD* for almost all cases tested.

Figure 7 shows solution CPU times for each model (standard and nested decomposition). The horizontal axes correspond to attacks and defensive resources. We can see how solution time increases as more resources come into play. Moreover, it can be observed that the amount of attack resource has remarkably more effect on solution time than does defensive resource. Finally, comparing both graphs, the smaller volume under the solution-time surface for the nested decomposition indicates, as also see from Table 1, a better overall performance.

Algorithm 1B proposed in Section IV.A proves to be faster (1.64 times on average) than the standard decomposition method (Table 10 in Appendix II). The tightening of the lower bound by solving a capacity-expansion LP is especially useful in the early iterations of the decomposition algorithm. For the first 6 problems tested (Table 10) Algorithm 1B performs on average 4.70 times faster than Algorithm 1, but not too

much improvement is obtained on the remaining 30 problems (1.02) when more than 10 iterations are typically needed to meet the algorithm's optimality criterion.

The next experiment explores the efficiency of Algorithm 2 (Section IV.B), when the same sample problems are solved by implementing the capacity-interdiction version of *DAD,* [DAD^{LP}-CN]. Table 2 shows a summary of the results for different values of "M".

		Algorithm 2 [DAD^{LP}-CN] ***Capacity-interdiction*** **(M=2)**				**Algorithm 2** [DAD^{LP}-CN] ***Capacity-interdiction*** **(M=1)**			
A	D	CPU time E (sec.)	Elap. time (sec.)	Iter.	Ratio A/E	CPU time F (sec.)	Elap. time (sec.)	Iter.	Ratio A/F
2	2	6.7	8.3	5	0.39	6.0	8.3	6	0.44
3	3	31.8	35.0	6	0.68	19.2	21.4	6	1.12
4	4	387.0	394.1	14	0.28	74.1	77.5	10	1.48
5	5	1363.0	1369.6	18	0.29	296.1	303.9	17	1.34
6	6	12620.0	12635.9	35	0.24	1799.0	1815.7	39	1.69
7	7	24210.0	24226.5	39	0.26	4717.0	4732.9	37	1.33
				Avg:	**0.45**			**Avg:**	**1.36**

Table 2. Computational results for Algorithm 2 (Section IV.B.8.). The "ratio" denotes the improvement (decline) of the algorithm with respect to the performance of Algorithm 1 implementing the standard decomposition *DAD* model. The value of M is a key issue in the *capacity-interdiction* model.

A careful selection of the value of M is required. A large value for M, such as M=2, is very expensive computationally speaking. As we see in the right hand side of the table, for M=1, Algorithm 2 is remarkably faster, close to one order of magnitude. In fact, its overall performance approaches that of the nested decomposition algorithm.

However, if M is not sufficiently large, the algorithm may give an incorrect solution. Among all the cases tested for M=1 (Table 11 in Appendix II), in five occasions there is a discrepancy in the objective-function value. This means that the selected value of M is not large enough for the algorithm to work correctly.

B. BOUND QUALITY FROM ADD MODELS

Although they solve different problems, the reordering-based *ADD* models have the ultimate objective of providing a valid bound for the general *DAD* problem. This section explores the quality of the bound and the execution times when the standard decomposition algorithm for *DAD* model (Algorithm 1) is compared to the general $[ADD^{+LP}]$ (Algorithm 3 in Section IV.C) and the specialized ADD^{+} model for DSP ($[MXSP^{+}]$, Section IV.D).

The network used in this test is the same 10×10 lattice used in the previous section. Note that the attacker always gets his original amount of resource plus the defender's resource: $b^x = b^x + \delta$, where $\delta = b^w$.

As we observe in the columns for the "relative differences" in Table 3, $[ADD^{+LP}]$ and $[MXSP^{+LP}]$ provide a reasonable lower bound for the objective value, although it seems to worsen when more resources are added to the problem.

		$[DAD^{LP}]$ **Algorithm 1**		$[ADD^{+LP}]$ **Algorithm 3**				$[MXSP^{+LP}]$			
A	D	z^*	CPU (sec)	$\underline{z}^*$	diff	Rel diff	CPU (sec)	$\underline{z}^*$	diff	Rel diff	CPU (sec)
2	3	4.58	3.4	4.58	0.00	0.0%	34.2	4.55	0.03	0.7%	44.0
2	4	4.58	4.1	4.53	0.05	1.1%	353.9	4.53	0.05	1.1%	1350.0
3	3	4.75	21.5	4.49	0.26	5.5%	104.8	4.55	0.20	4.2%	540.5
3	4	4.75	32.9	4.58	0.17	3.6%	477.6	4.55	0.20	4.2%	190.8
4	2	4.96	66.6	4.59	0.37	7.5%	80.0	4.58	0.38	7.7%	879.1
5	2	5.20	150.8	4.59	0.61	11.7%	321.8	4.59	0.61	11.7%	3621.0
5	3	5.07	211.0	4.59	0.48	9.5%	4916.0	4.58	0.49	9.7%	5632.0
					Avg	**5.5%**			**Avg**	**5.6%**	

Table 3. Bound quality for ADD^{+} models solving DSP. "A" and "D" denote attack and defensive resources respectively; z^* denotes the objective-function value of *DAD* model for the related problems; $\underline{z}^*$ denotes the bound provided by ADD^{+} models; and the "diff" column represents the difference between z^* and $\underline{z}^*$.

As proposed in Section III.B, we would like to use the solutions provided by both [ADD^{+LP}] and [$MXSP^{+LP}$], as a strong bound for *DAD*. Furthermore, by means of solution-elimination constraints, we could approximate the bound to the objective-function value to meet the optimality criterion. However, as shown in Table 3, the solution times at the same level of tolerance, are not comparable: those for the standard decomposition *DAD* model are one order of magnitude faster than those for [ADD^{+LP}] and [$MXSP^{+LP}$].

Algorithm 4, presented in Section IV.D, proposed a decomposition method [$MXSP^{+LP}$-D] to solve [$MXSP^{+LP}$]. The former proves to be faster especially for networks with square topology as in this case (See results in Apendix II). In fact, when we run the same problems of Table 3, [$MXSP^{+LP}$-D] is 2.56 times faster than [$MXSP^{+LP}$], on average. However, solution times are still too long compared to those of standard *DAD* (Algorithm 1).

VI. PRACTICAL EXAMPLE

This chapter illustrates an example based on a hypothetical emergency deployment of a unit of the Spanish Marine Corps. This exercise requires the solution of an instance of DSP (Defending the Shortest Path) to plan defenses for a small Infantry entity that needs to traverse from its home base in San Fernando to the Naval Base in Rota, for emergency deployment. All the information used in this example has been gathered from open sources, such as the Internet, journals published by regional traffic management authorities, and the Spanish Department of Defense.

A. PROBLEM DEFINITION

The home base of the Spanish Marine Corp Brigade is located in the city of San Fernando, province of Cadiz, in the south of Spain. The city has a population of over 90,000 (Wikipedia 2007), is expanding rapidly and the brigade's movements just outside the base can easily become entangled in the consequent construction activity. This fact is important in this scenario: The brigade needs to reach the Naval Base located in Rota, thirty miles north of San Fernando at the opposite side of Cadiz Bay (Figure 8).

The Special Operation Forces (SOF) are also located in the same military installations as the Brigade.

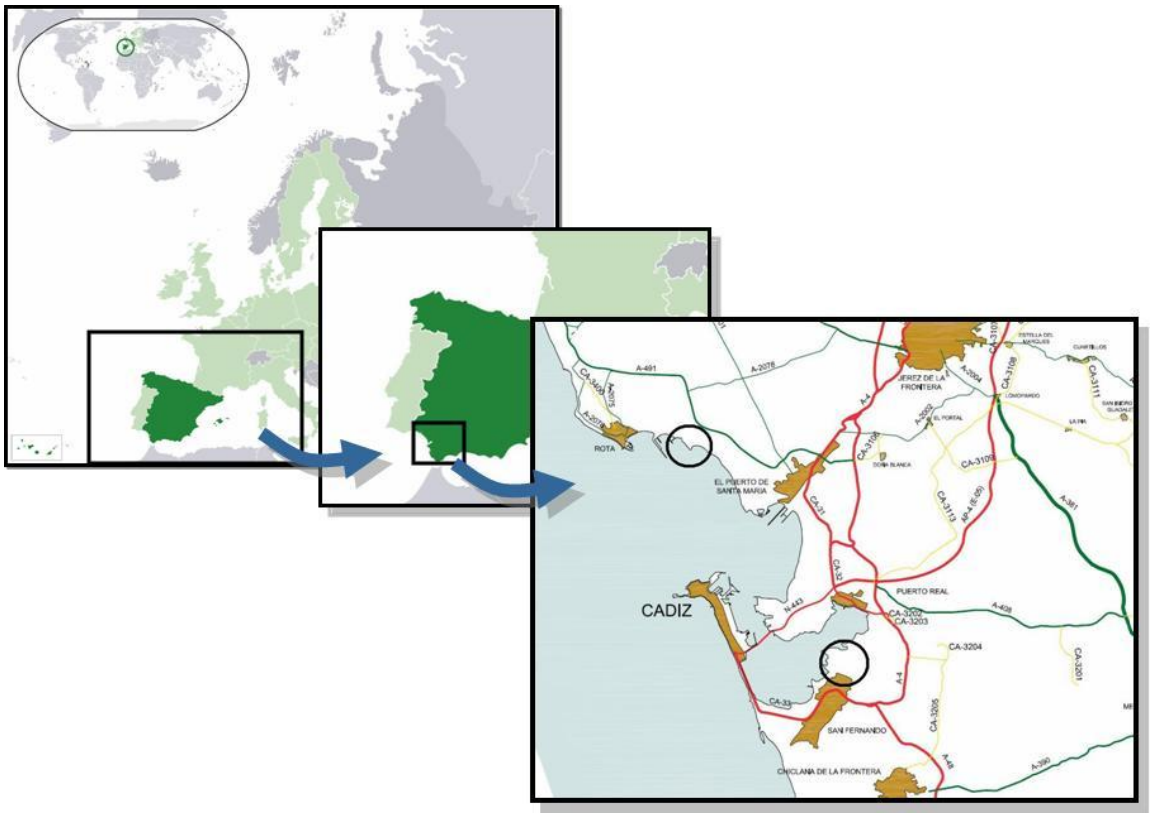

Figure 8. Map of Cadiz Bay (Spain) showing the two sites of interest (Map from Wikipedia 2007).

Let us consider the following hypothetical scenario: A frigate from the 41st Fleet Squadron (home-based at Rota Naval Base) is alerted and receives orders to get underway immediately for a maritime interdiction operation against a suspect vessel transiting through the Strait of Gibraltar. For that mission, an *estol* (a small special operations unit) is also alerted, and ordered to transit to the dock in Rota Naval Base, prior to the frigate's departure. Since air assets are already committed to other tasks, the *estol* must transit using its own means, which means with humvees and trucks. The suspect vessel is operated by a terrorist organization that soon becomes aware that the most likely spot for the vessel to be interdicted is precisely in the Strait (in Spanish territorial waters). Thus, a terrorist dormant cell is alerted to conduct counter-deployment actions.

Since the *estol* (who plays the role of the "defender-operator") constitutes a small and indivisible infantry unit, we will model its transit to destination as a shortest-path problem: the *estol* must transit from San Fernando (source node) to Rota (sink node) in the minimum possible time.

Continuing with the scenario, the now-active terrorist cell, i.e., the "attacker," comprises six autonomous units with enough striking power to put road segments out of action for at least one hour. We might envision an attack as a chemical spillage caused by a deliberate wreck of a previously hijacked truck. This truck transports hydrogen cyanide, which is a widely used agent in many industrial processes and has a persistency in soil close to one hour (Sidell 2002).

By means of intelligence reports, this information is known by the *estol's* Special Operations Forces command, i.e., the "defender." SOF command has the option to plan ahead, which means deploying up to 10 patrolling units along the road network. To minimize the maximum transit delay the terrorist cell can achieve, SOF command must solve a tri-level optimization problem to use sparse resources wisely. This and similar scenarios would probably be foreseen, however, and the solution to this problem would be available "off the shelf."

Further assumptions for the problem are:

- By traversing an interdicted road, the *estol* "agrees" to pay the penalty in its entirety. The full delay is incurred if the road segment is traversed, regardless of the time it takes the *estol* to get there. (We are being conservative with respect to the *estol's* transit time, because some cleanup might have been completed by the time that the *estol* reaches a section of road that has been attacked.)
- At time zero, when the *estol* starts its mission, all transit assets (i.e., vehicles and trucks) are available for immediate use.
- The model does not account for transit delays in road intersections.

B. BUILDING THE NETWORK

Given the background above, the next step is to develop a model of the relevant road network. This is facilitated through road maps of the Cadiz Bay area, information provided by local authorities regarding traffic routes (Consejería de Obras Públicas y Transportes 2006), and satellite images downloaded from the Internet (Google 2007). A total of 195 nodes are identified. Table 4 shows a snapshot of the list containing all nodes, each with a brief description, the city or county they belong to, geographical position in latitude and longitude coordinates, and the node type (with 1 being the source, –1 the sink, and 0 indicating a transit node). Appendix III presents the full table.

Node number	description	Location	Latitude	Longitude	Type
1	TEAR	San Fernando	36º 28'43.83" N	6º 11'30.05" W	1
2	Armada & la Clica	San Fernando	36º 28'51.96" N	6º 11'28.67" W	0
3	La Carraca bridge	San Fernando	36º 28'49.28" N	6º 10'50.59" W	0
4	La Carraca dock	San Fernando	36º 20'54.69" N	6º 10'51.59" W	0
5	Arapiles rd & La Carraca rd	San Fernando	36º 28'43.55" N	6º 11'02.54" W	0
6	Fadricas rd & Caserio de Ossio	San Fernando	36º 29'00.20" N	6º 11'45.00" W	0
7	Fadricas rd & Magallanes	San Fernando			0
8	Sfdo Train Station	San F			
	Magallanes & Ferrocaril				

Table 4. Node-list snapshot for the sample network. It includes node description, location, geographical position and type (type "1" represents the source node, "0" a transit node and "−1" the sink node (not shown here).

Since our question revolves around response times, the obvious cost on each arc will represent the nominal time a vehicle takes for a one-way transit. To find this time, we require two values for each arc: distance and speed. By dividing distance by speed, we obtain a transit time, hereafter referred to as "cost."

The distance values are found using maps and online navigation engines such as Google 2007. Finding speed values is not so straightforward, necessitating some subjective estimates. We develop a ranking scale of one to five representing five distinct average transit speeds, ranging from 15 mph to 55 mph, in intervals of 10 mph. Each route segment is assigned one of these values, based on the type of road (e.g., highway, freeway, local road) Also, other factors are taken into account such as the number of

lanes, the state of pavement, number of intersections, degree of straightness, etc. A total of 632 segments are labeled encompassing 453.4 miles of road.

Finally, each arc of the network needs to be assessed in terms of amount of delay that an interdiction might cause. Two aspects are considered here: persistency of the chemical agent and the shortest distance (minimum time) to the nearest Civil Protection or Emergency Management Centers (we assume that setup procedures, safety protocols and cleanup tasks are implemented as soon as the first emergency-response team reaches the scene of the "chemical attack" and the clock starts running). The nine centers are located in the area map and the shortest distances from each one of them to every other node in the network are calculated.

Let $\mathcal{N} = \{1, 2, ..., 207\}$ denote the set of all nodes in the network and $\mathcal{C} = \{1, 2, ..., 9\}$ denote the set of all emergency centers. Also, let $g(c, n)$ denote the minimum cost (transit time) from emergency center $c \in \mathcal{C}$ to node $i \in \mathcal{N}$. Then, the delay coefficient $d_{i,j}$ for the road segment $(i, j) \in A$ is calculated as:

$$d_{i,j} = p_a + \frac{1}{2}\left(\min_{c \in \mathcal{C}} \left(g(c,i) \right) + \max_{c \in \mathcal{C}} \left(g(c,j) \right) \right)$$

where p_a denotes the persistency of the chemical agent. The second term represents the average of response times to the head and tail node of the arc (i, j), in case this arc is attacked. We are assuming that emergency centers have enough capacity to provide response teams for all possible incidents, and that response teams are not themselves subject to delays. (Response teams have protective equipment and can pass easily through one attack site to reach another.)

The following table shows a snapshot of the list containing, for every arc, its transit-speed rank, length (miles), transit-speed (mph), cost (hours), and penalty if attacked (hours):

From	to	ranked value	distance	speed	cost	delay
1	2	2	0.11	25	0.004	1.237
1	8	1	0.39	15	0.026	1.233
1	32	1	0.38	15	0.025	1.233
2	1	2	0.11	25	0.004	1.237
2	3	3	0.68	35	0.019	1.237
2	6	2	0.6	25	0.024	1.246
2	31	2	1.17			
3	2	3				

Table 5. Snapshot of the arc list for the sample network. It includes tail and head nodes, transit-speed rank, distance (miles), transit-speed (miles/hour), cost (hours) and delay (hours).

Appendix III provides a complete listing of the final network data.

C. SOLVING THE PROBLEM

Algorithm 1, based on $[DAD^{LP}mxm]$, is implemented in GAMS (GAMS Development Corporation 2007) with the following runtime parameters:

- Solver for LPs and MIPs: XPRESS (v. 16.10)
- Absolute and relative termination criterion for MIP: 0.0
- Allowable relative gap between bounds in the Benders decomposition method: 0.0

Computation is performed on a personal computer (Processor x86, Family 6, Model GenuineIntel 1596 Mhz with 1 Gb of RAM). Table 6 shows the results for this problem:

Estol deployment example. Summary 1 $[DAD^{LP}]$		
CPU time	9,112.0	sec.
Num. variables in master problem	633	
Num. of iterations in the algorithm	184	
Shortest s-t path length with no attacks	0.5288	hrs
Shortest s-t path length with 6 attacks, no defenses	2.7356	hrs
Shortest s-t path length with 6 attacks, 10 defenses	1.7019	hrs

Table 6. Computational results for Algorithm 1 $[DAD^{LP}]$. The number of equations in the final master problem comprises 184 "cuts" plus one resource constraint.

The complete optimal defense plan is listed in Table 13. As anticipated, the defender-operator (*estol*) does not use all the road segments that have been defended $\left(w_k = 1 \not\Rightarrow y_k = 1\right)$. The reason is that the SOF Command, lacking defensive resources to protect one route entirely, seeks to spread the patrolling effort among the main three routes out of San Fernando city. With six units of resource, the terrorist group is able to interdict each of those, achieving the group's goal of delaying the *estol's* transit to its destination. Figure 12 depicts this solution over the area map.

Next, we proceed to test the other algorithms to see how well they perform on this problem. Table 7 shows the results obtained when applying $[DAD^{LP}$-CN2] (see equations (18)-(21)).

Estol deployment example. Summary 2 $[DAD^{LP}$-CN2]***, (M=2)***		
CPU time	43,000	sec.
Num. variables in master problem	633	
Num. of iterations in the algorithm	49	
Shortest s-t path length with no attacks	0.5288	hrs
Shortest s-t path length with 6 attacks, 10 defenses	1.7019	hrs

Table 7. Computational results of Algorithm 2 $[DAD^{LP}$-CN2] with M = 2.0

The algorithm yields the same solution, but it takes almost 12 hours to solve. The burden of the algorithm is the value of the M coefficient. Consequently, different values

of M are tried: For M = 1, the problem only takes 3,085 seconds (51 min.) but the objective value that it yields (1.6361) is 3.9% smaller than what it should be. With $M = 1.2 = \max_k d_k$, Algorithm 2 finds the optimal solution (and proves optimality) in a more reasonable completion time of 3 hours and 9 minutes.

Implementing $[ADD^{LP}]$ to compute an optimistic lower bound for this problem leads to a dead end. Since the defender has more resources (10) than the attacker (6), the former is able to nullify all the attacks and always bring down the lower bound to the value of the shortest s-t path with no attacks.

On the other hand, the decomposition algorithm for $[ADD^{+LP}]$ (Algorithm 3 in Section IV.C), with 16 units of resource for the attacker, moves its bounds sluggishly toward the optimal value of the problem. After 10 hours of execution, the lower bound is still only 0.58 (and not close to proving optimality because the global upper bound is 4.11) versus a potential value as large as $z^* = 1.709$. Since the *d* coefficients are a critical parameter in *ADD* models, we apply a systematic reduction of 0.5 hours to all road segments, hoping that the solution is still valid and the bounds converge more quickly. However, the convergence of the bounds is still too slow.

Finally, as an alternative way to solve $[ADD^{+LP}]$ and to obtain an optimistic lower bound on z^*, the specialized model $[MXSP^{+LP}]$ is run with 16 units of resource for the attacker and 10 for the defender. The problem solves in under 3 hours but yields an optimal objective value of 0.53, which still is too far from $z^* = 1.709$ to be of any value.

D. ANALYSIS

Because of the particular structure of the road network for this deployment problem, the terrorist group can, with six strikes, disrupt all main routes that connect San Fernando and Rota. The *estol* has no other alternative than to wait for the completion of at least one cleanup, which, unavoidably, delays its transit to the Naval Base.

A quick look at the solution shows that attacks are spread throughout the network and not concentrated around the source and sink nodes. The reason for that might be that the attacker gains a little more reward by placing his attacks far from emergency-response centers.

Algorithm 1 seems to be the most useful method to solve DSP for this scenario. In addition, Algorithm 2 [*DAD*-CN] offers a reasonable alternative, provided that that the constant M is selected carefully.

The fact that the penalty coefficients are much greater than the costs (30.52 times larger on average), makes the problems difficult to solve (see Israeli and Wood 2002). It becomes clear in the implementation of $[ADD^{+LP}]$, where the time required to solve (11+ hours) is unacceptable for practical purposes. Therefore, those coefficients need to be reduced or tightened in such a way that they remain valid for the original problem. When we try this technique on Algorithm 1 $[DAD^{LP}]$, by artificially reducing all d_k coefficients by 0.50 (the smallest penalty is still larger than the shortest s-t path with no attacks), we obtain the correct solution and, interestingly, a remarkable reduction in execution time (4,359 seconds).

However, the d_k coefficient-reduction technique does not seem to work with $[ADD^{+}]$.

THIS PAGE INTENTIONALLY LEFT BLANK

VII. CONCLUSIONS

This thesis has defined and developed tri-level models to solve the problem of defending critical infrastructure. The objective of the *defender* in the outer level of these models is to protect a set of activities given a certain amount of defensive resource. The goal is to minimize the worst damage that a potential *attacker* can inflict to the system. Damage is measured by increased costs at the innermost level, assuming that the *defender-operator* operates the system optimally. Of course, "cost" can represent unsatisfied demands, time delays in achieving goals, and minimizing "cost" can also represent maximizing effectiveness, e.g., system output.

We have formulated a *DAD* model that is, apparently, solvable only through decomposition. Algorithm 1 is devised to solve the *DAD* model. The master problem of this decomposition algorithm looks like a master problem for a standard Benders decomposition of a mixed-integer linear program, but the subproblem is a mixed-integer linear program (MIP) rather than standard linear program (LP). Algorithm 1 solves this MIP directly, i.e., with LP-based branch-and-bound, while Algorithm 1A solves the MIP by Benders decomposition, and thus, it may be viewed as a nested decomposition algorithm. Algorithm 1A proves to be almost twice as fast as Algorithm 1 (1.87 times faster, on average) on test problems that involve "defending the shortest path" (DSP). DSP represents a situation in which the defender-operator needs to solve an A-B shortest-path problem to operate the system optimally, that is, he wants to move from node A in a road network to node B in the minimum time possible; the attacker seeks to maximize this minimum path length by interdicting a limited number of road segments and making them impassible (or adding a delay to their traversal times); but before any attacks occur, the defender can make a limited number of road segments invulnerable to attack. against a limited number of attacks so as to reduce effects. The defender's goal is to minimize the maximum shortest-path length.

For a small number of offensive and defensive resources, the overhead of a nested decomposition algorithm suggests the use of the standard decomposition method (Algorithm 1). In addition, for the size of the shortest-path network, the factor "available

attack resource" proves to affect solution time more than the amount of defensive resource, specifically, more attack resource leads to longer solution times.

Algorithm 1B is similar to Algorithm 1 except for the inclusion of a *capacity-expansion* LP to tighten the lower bound. This proves to be especially useful on the early iterations of the algorithm. In the cases where the numbers of attacks are sparse and the algorithm typically tends to solve with few iterations, 1B is, on average, 4.70 faster than Algorithm 1 for the DSP on a square lattice.

A different approach to the tri-level problem is the reformulation-based *capacity-interdiction* model [*DAD*-CN]. This formulation is somewhat more complex than the general *DAD* alluded to above and is more difficult to implement. It too requires a decomposition algorithm to solve (Algorithm 2), and it proves to be almost as fast as Algorithm 1A. Its solution times depends heavily on a "big-M" value used in linearizing the model. An excessively large value weakens the subproblem and leads to a poor performance in terms of completion time. On the other hand, a too small value of M speeds up the algorithm but does not guarantee a correct solution. It might be interesting for future research to investigate useful techniques to tighten the upper bound coming from the decomposition subproblem by selecting an appropriate value of M.

Interchanging the order of the first two levels of optimization in the model, that is, converting min-max-min into min-min-max, can provide an optimistic (lower) bound on the optimal *DAD* objective-function value. We give the advantage to the defender, who sees the attacker's plan before defending the system and operate it. The quality of the bound provided by this reordering-based *ADD* might be poor, however, but we have shown that it is possible to improve the bound by adding appropriately to the attacker's resource to create "ADD^+." The relative differences between the bound obtained from ADD^+ and the optimal value of *DAD* are reasonably small (5.5% on average) when the DSP problem is implemented on a square lattice, although these differences tend to increase as resources for both attacker and defender increase together. We would like to use this bound in the solution of *DAD*. However, the computational times observed for ADD^+ are excessive, some times larger than solving *DAD* by, say, Algorithm 1.

Interestingly, for DSP, ADD^+ can also be solved as a "Maximizing the Shortest-Path" (*MXSP*) in an expanded network, assuming that the attacker and defender are constrained only by the number of arcs that can be attacked or defended respectively. *MXSP* is a bi-level attacker-defender network-interdiction model. The original network is expanded in levels according to the number of arcs that can be defended. *MXSP* can be solved directly or by using Benders decomposition. The latter method proves to be faster than ADD^+ itself, but it is still too slow compared to *DAD*, at least for DSP.

A practical DSP example is presented to illustrate the effectiveness of the models and solution procedures to solve realistic problems. Essentially, a small Spanish infantry unit must traverse from the Marine Corp HQ to the Naval Base for immediate deployment, using the road network in Cadiz Bay (Spain). A terrorist group is able to interdict up to 6 road segments, and the Marine Corp Command has the option to plan ahead, protecting 10 segments by means of armed patrols. The network built to represent this example contains almost 200 nodes and 630 arcs. The problem is solved by Algorithm 1, implemented in GAMS, in a reasonable execution time of two and a half hours.

The fact that the penalty coefficients are much greater that the costs makes the problems difficult to solve. Thus, a systematic reduction in all coefficients was made in hopes that the solution obtained is still valid. As a precautionary measure, no penalty should decreased below the value of the shortest s-t path without attacks. The procedure in this case proves to be valuable, cutting down the execution time by half.

In future research, it may be interesting to explore other instances of the tri-level problem, perhaps one in which the theory developed here regarding *ADD,* applies better than the DSP.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX I. NOTATION

The following describes the notation and major symbols used in this thesis. Particular cases may need additional notation and this is explained when used. Vectors, represented by lower-case bold letters, are column vectors. Uppercase letters represent matrices, and Greek letters dual variables.

Symbol	Description
$\mathbf{c}$	Operating cost
D	Diagonal matrix of penalties
f	Function, usually in an objective (e.g., $\max f(x)$)
g	Function, usually in an objective (e.g., $\min g(x)$)
i, j	Network nodes
k	System activities
t	Superscript index for decomposition methods
U	Diagonal matrix of activity capacities
v	Binary variable used in the practical example
$\mathbf{w}$	Defensive plan (binary vector)
W	Feasible set for defensive plans $\mathbf{w}$
$\mathbf{x}$	Attack plan (binary vector)
X	Feasible set for attack plans $\mathbf{x}$
$\mathbf{y}$	Defender-operator's operating plan
Y	Feasible set for operating plans $\mathbf{y}$
z	Objective value of an optimization model

Symbol	Description
$\boldsymbol{\alpha}$	Dual variables for operating constraints
$\boldsymbol{\beta}$	Dual variables for capacity-interdiction constraints
δ	Real number
ε	Real number that usually denotes tolerance
$\boldsymbol{\varphi}$	Dual variables for railroad constraints (practical example)
$\boldsymbol{\theta}$	Dual variables for capacity constraints
$\boldsymbol{\pi}$	Dual variables for operating constraints
$\varnothing$	Empty set
$\sum$	Summation
*	Optimal value or solution (e.g., z^*)

Table 8. Notation and definition of terms used.

APPENDIX II. DETAILED COMPUTATIONAL RESULTS FOR PROBLEMS IN CHAPTER V

A. STANDARD DAD AND NESTED DECOMPOSITION

			Standard decomposition Algorithm 1			**Nested decomposition Algorithm 1A**			**Ratio**
Problem	**Att.**	**Def.**	Time A (sec)	Elap (sec)	Iter	Time B (sec)	Elap Time	Iter	A/B
1	2	2	2.7	4.2	3	1.2	7.5	3	2.30
2	2	3	3.4	5.3	4	3.3	13.2	5	1.04
3	2	4	4.1	6.4	5	3.1	17.6	7	1.32
4	2	5	4.3	7.0	6	3.9	21.5	9	1.12
5	2	6	7.5	12.1	10	4.5	23.2	10	1.66
6	2	7	7.8	13.9	12	7.3	32.0	15	1.06
7	3	2	17.4	20.7	5	12.2	38.3	6	1.43
8	3	3	21.5	24.8	7	10.7	31.0	6	2.01
9	3	4	32.9	39.2	13	27.5	68.8	11	1.20
10	3	5	39.9	48.5	17	29.9	74.6	14	1.34
11	3	6	44.4	55.4	21	15.7	53.0	13	2.82
12	3	7	65.6	81.8	34	29.1	107.6	22	2.25
13	4	2	64.7	68.2	7	38.2	75.9	6	1.69
14	4	3	93.1	98.3	10	69.8	128.3	10	1.33
15	4	4	109.6	114.8	10	57.3	124.9	13	1.91
16	4	5	183.1	192.9	21	84.9	191.2	17	2.16
17	4	6	240.5	254.3	30	75.6	177.7	21	3.18
18	4	7	287.0	303.7	37	151.4	342.5	35	1.90
19	5	2	150.8	157.6	6	111.0	172.5	8	1.36
20	5	3	211.0	215.2	8	155.9	235.6	11	1.35
21	5	4	473.6	480.2	21	238.8	397.3	16	1.98
22	5	5	396.6	406.4	20	306.0	484.2	27	1.30
23	5	6	548.2	562.8	30	337.6	554.2	30	1.62
24	5	7	682.6	689.9	33	362.7	579.0	26	1.88
25	6	2	852.8	856.3	7	394.3	467.0	6	2.16
26	6	3	979.3	983.2	8	691.5	816.7	12	1.42

			Standard decomposition Algorithm 1			Nested decomposition Algorithm 1A			Ratio
27	6	4	1587.0	1594.6	16	1030.0	1227.3	18	1.54
28	6	5	2244.0	2255.8	26	1210.0	1451.8	25	1.85
29	6	6	3047.0	3067.6	42	1719.0	2075.9	31	1.77
30	6	7	4331.0	4362.2	69	1530.0	1897.3	36	2.83
31	7	2	389.7	393.0	7	698.4	737.2	4	(0.56)
32	7	3	956.8	962.8	13	166.5	175.0	2	5.75
33	7	4	1633.0	1643.4	22	617.2	648.0	2	2.65
34	7	5	2212.0	2223.7	25	2302.0	2417.1	9	(0.96)
35	7	6	2942.0	2959.3	39	5625.0	2500.1	36	(0.52)
								Avg:	**1.87**

Table 9. Computational results for Algorithm 1 and Algorithm 1A implementing DSP with the standard decomposition and the nested decomposition methods, respectively Grid: lattice 10×10. Number of decision variables: 383. Costs range: Uniform [0,1]. Penalties range: Uniform [1,2]. Model implementation: GAMS. Solver: XPRESS. Other parameters: optcr=0.01, allowable decomposition gap=0.01. The "ratio" column corresponds to the improvement (decline) of Algorithm 1A with respect to Algorithm 1.

B. IMPROVED STANDARD DAD DECOMPOSITION

				Standard decomp. Algorithm 1		Algorithm 1B		Ratio
Prob.	**Att.**	**Def.**	z^*	time A (sec)	Iter.	time C (sec)	Iter.	A/C
1	2	2	4.58	2.7	4	1.0	1	2.63
2	2	3	4.58	3.4	5	1.0	1	3.37
3	2	4	4.58	4.1	6	1.0	1	4.02
4	2	5	4.58	4.3	7	1.0	1	4.21
5	2	6	4.55	7.5	11	1.1	1	6.86
6	2	7	4.55	7.8	13	1.1	1	7.14
7	3	2	4.78	17.4	6	17.4	5	1.00
8	3	3	4.75	21.5	8	21.5	7	1.00
9	3	4	4.75	32.9	14	33.0	13	1.00

Prob.	Att.	Def.	z^*	time A (sec)	Iter.	time C (sec)	Iter.	A/C
10	3	5	4.75	39.9	18	41.4	17	0.96
11	3	6	4.58	44.4	22	40.7	18	1.09
12	3	7	4.58	65.6	35	41.6	21	1.58
13	4	2	4.96	64.7	8	64.1	7	1.01
14	4	3	4.93	93.1	11	94.4	10	0.99
15	4	4	4.81	109.6	11	109.9	10	1.00
16	4	5	4.79	183.1	22	184.0	21	1.00
17	4	6	4.79	240.5	31	241.0	30	1.00
18	4	7	4.78	287.0	38	294.2	33	0.98
19	5	2	5.20	150.8	7	150.8	6	1.00
20	5	3	5.07	211.0	9	208.7	8	1.01
21	5	4	5.07	473.6	22	480.1	21	0.99
22	5	5	5.07	396.6	21	399.3	20	0.99
23	5	6	5.02	548.2	31	557.1	30	0.98
24	5	7	5.02	682.6	34	639.2	33	1.07
25	6	2	5.35	852.8	8	836.0	7	1.02
26	6	3	5.28	979.3	9	963.0	8	1.02
27	6	4	5.25	1587.0	17	1591.2	16	1.00
28	6	5	5.25	2244.0	27	2493.0	24	0.90
29	6	6	5.18	3047.0	43	3251.0	40	0.94
30	6	7	5.12	4331.0	70	4041.0	59	1.07
31	7	2	5.68	389.7	8	381.7	7	1.02
32	7	3	5.64	956.8	14	953.0	13	1.00
33	7	4	5.59	1633.0	23	1554.0	20	1.05
34	7	5	5.53	2212.0	26	2473.0	25	0.89
35	7	6	5.35	2942.0	40	3670.0	39	0.80
							AVG:	1.64

Table 10. Computational results for modified Algorithm 1B implementing DSP. (See Table 9 for problem and implementation parameters). The "ratio" column corresponds to the improvement (decline) of Algorithm 1B with respect to Algorithm 1.

C. CAPACITY-INTERDICTION DAD

			Capacity-interdiction *(M=2)* Algorithm 2				**Capacity-interdiction (M=1) Algorithm 2**			
Prob	**Att.**	**Def.**	time E (sec)	Elap time	Iter.	A/E	time F (sec)	Elap time	Iter.	A/F
1	2	2	6.7	8.3	4	(0.39)	6.0	8.3	5	(0.44)
2	2	3	5.8	7.5	4	(0.58)	3.4	4.5	3	(0.99)
3	2	4	8.6	11.0	5	(0.48)	4.4	6.2	5	(0.95)
4	2	5	11.0	13.7	7	(0.40)	8.5	11.8	9	(0.51)
5	2	6	10.2	12.9	7	(0.73)	6.2	9.1	8	1.20
6	2	7	10.7	14.3	8	(0.73)	7.5	12.1	10	1.04
7	3	2	37.0	38.9	5	(0.47)	15.5	17.4	4	1.12
8	3	3	31.8	35.0	5	(0.68)	19.2	21.4	5	1.12
9	3	4	27.1	28.9	4	1.21	16.8	20.0	5	1.96
10	3	5	47.0	51.0	9	(0.85)	29.5	34.1	10	1.35
11	3	6	59.4	65.1	13	(0.75)	30.8	36.3	12	1.44
12	3	7	62.5	68.7	15	1.05	26.0	31.2	12	2.52
13	4	2	142.6	145.0	6	(0.45)	36.1	38.4	5	1.79
14	4	3	204.1	206.7	7	(0.46)	94.6	98.1	8	(0.98)
15	4	4	387.0	394.1	13	(0.28)	74.1	77.5	9	1.48
16	4	5	272.6	278.3	11	(0.67)	128.8	136.0	15	1.42
17	4	6	288.0	295.6	16	(0.84)	125.4	134.5	17	1.92
18	4	7	315.5	323.9	21	(0.91)	103.3	111.3	15	2.78
19	5	2	809.4	811.8	6	(0.19)	133.3	137.5	7	1.13
20	5	3	1059.0	1062.8	10	(0.20)	184.7	189.2	10	1.14
21	5	4	1211.0	1215.9	11	(0.39)	225.8	231.3	13	2.10
22	5	5	1363.0	1369.6	17	(0.29)	296.1	303.9	16	1.34
23	5	6	1184.0	1192.0	20	(0.46)	568.7	580.7	25	(0.96)
24	5	7	1962.0	1976.9	37	(0.35)	588.5	602.7	36	1.16
25	6	2	4237.0	4239.4	5	(0.20)	424.8	426.9	5	2.01
26	6	3	6988.0	6992.6	10	(0.14)	776.3	780.1	10	1.26
27	6	4	7320.0	7325.3	12	(0.22)	1114.0	1121.3	18	1.42
28	6	5	12440.0	12455.6	24	(0.18)	1157.0	1165.7	21	1.94
29	6	6	12620.0	12635.9	34	(0.24)	1799.0	1815.7	38	1.69

Prob	Att.	Def.	time E	Elap	Iter.	A/E	time F	Elap	Iter.	A/F
30	6	7	14620.0	14588.9	40	(0.30)	1565.0	1581.1	38	2.77
31	7	2	3430.0	3431.9	5	(0.11)	371.2	374.4	7	1.05
32	7	3	5180.0	5197.4	7	(0.18)	1570.0	1573.7	10	(0.61)
33	7	4	11950.0	11959.8	17	(0.14)	2985.0	2991.9	17	(0.55)
34	7	5	13830.0	13839.7	20	(0.16)	2477.0	2436.2	21	(0.89)
35	7	6	14440.0	14447.5	24	(0.20)	5180.0	5194.4	34	(0.57)
36	7	7	24210.0	24226.5	38	(0.26)	4717.0	4732.9	36	1.33
					Avg:	**(0.45)**			Avg:	**1.36**

Table 11. Computational results for Algorithm 2 [DAD^{LP}-CN2], implementing DSP for different defensive and interdiction resources. The improvement in solution times with respect to the standard decomposition [DAD^{LP}] is given by the ratio column (See Table 9 for problem and implementation parameters).

D. MXSP PROBLEM IN AN EXPANDED NETWORK MODEL

This section of the appendix illustrates the results obtained for the specialized *ADD* model developed in the body of the thesis, in Section D.IV.

Table 12 shows execution times, in seconds, when [$MXSP^{LP}$] and [$MXSP^{LP}$-D] are tested against a set of networks of different topology and increasing size. The long topology networks are rectangular grids with 3 nodes in the vertical axis and $m > 3$ nodes in the horizontal axis. Similarly, square-topology networks are based on an $n \times n$ grid nodes. They are generated using Java code that randomly assigns costs to the arcs within the range [0, 1] and interdiction penalties in the range [1, 2].

The number of defense and attack resources are kept fixed throughout the runs to two and four, respectively. The XPRESS solver is used within GAMS (XPRESS Solver Manual, GAMS 2007), with zero absolute termination criterion and a value for the allowable decomposition gap of $\varepsilon = 0.1$ for the decomposition algorithm.

Problem	topology	nodes	interdictable arcs	decision vars G+	[$MXSP^{+LP}$]	[$MXSP^{LP}$-D]	iterations
1	3×3	11	24	168	0.08	0.17	3
2	3×5	17	44	308	0.25	1.76	10
3	3×7	23	64	448	0.44	1.97	18
4	3×10	32	94	658	1.61	28.60	90
5	3×13	41	124	868	2.53	15.30	47
6	3×15	47	144	1008	9.69	59.03	94
7	3×20	62	194	1358	5.31	29.16	53
8	2×25	77	244	1708	44.70	219.43	167
9	3×30	92	294	2058	13.94	61.87	74
10	3×50	152	494	3458	1342.00	462.25	162
11	4×4	18	48	336	0.16	0.77	10
12	5×5	27	80	560	0.63	1.10	10
13	6×6	38	120	840	2.50	3.77	26
14	7×7	51	168	1176	1.92	4.38	30
15	8×8	66	224	1568	45.33	13.20	55
16	10×10	102	360	2520	36.22	11.90	43
17	12×12	146	528	3696	47.91	52.50	120
18	15×15	227	840	5880	268.77	53.65	61
19	20×20	402	1520	10640	10130.00	300.25	131

Table 12. Computational results for MXSP showing execution times when DSP is solved directly as a MIP [$MXSP^{LP}$] or by decomposition [$MXSP^{LP}$-D], (See Algorithm 4 in Section IV.D).

Figure 9 and Figure 10 show that there is a small difference in computational times for different network topologies. Square grids tend to solve faster than long networks, when comparing grids with roughly the same number of nodes.

With respect to solution methods, the number of decision variables in the problem is the most significant factor. For grids with few nodes, [$MXSP^{LP}$-D] solves faster, but beyond 100 nodes, decomposition methods do much better. There are orders of magnitude in the differences between solution times when the number of nodes increases.

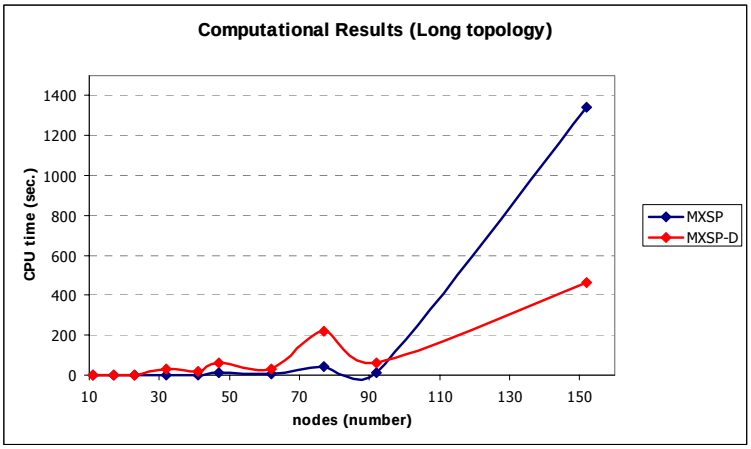

Figure 9. Computational results for long topology grids. Differences in performance between [$MXSP^{LP}$] and [$MXSP^{LP}$-D] become noticeable beyond 100 nodes (10×10 lattice).

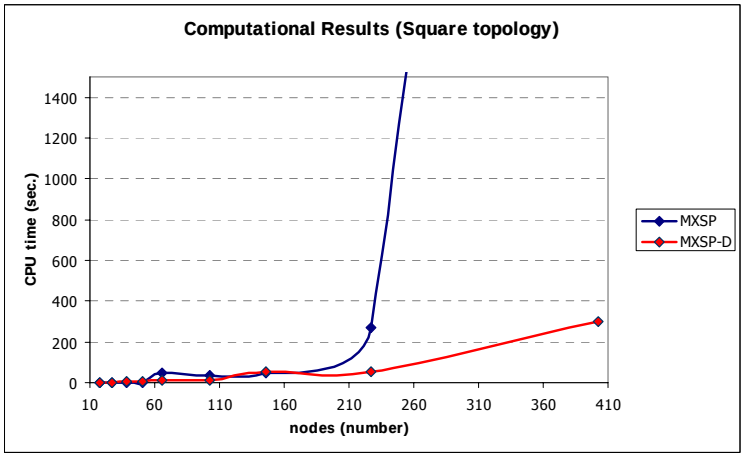

Figure 10. Computational results for square-topology grids. Beyond 200 nodes (15×15 lattice), solving [$MXSP^{LP}$] directly as a MIP is very computational expensive.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX III. PRACTICAL EXAMPLE IN CHAPTER VI: FIGURES AND TABLES

Figure 11. Cadiz Bay road map showing the network nodes. (Map from Michelin 2007).

PLANS FROM OPTIMAL SOLUTION							
Defensive plan		**Attack plan**		**Shortest path**			
tail	head	tail	head	tail	head	tail	head
1	2	50	51	1	8	122	140
36	50	50	52	8	14	123	125
55	57	107	106	14	16	125	122
57	70	108	105	16	27	126	123
70	72	158	171	27	110	140	141
97	120	169	170	90	92	141	145
98	97			92	93	145	146
105	184			93	121	146	149
120	127			94	95	149	203
121	126			95	97	173	174
				97	90	174	176
				100	94	175	173
				102	100	176	202
				105	184	177	300
				108	105	184	102
				109	108	202	177
				110	109	203	175
				121	126		

Table 13. Optimal defensive, attack and traversing plan for the *estol* DSP problem.

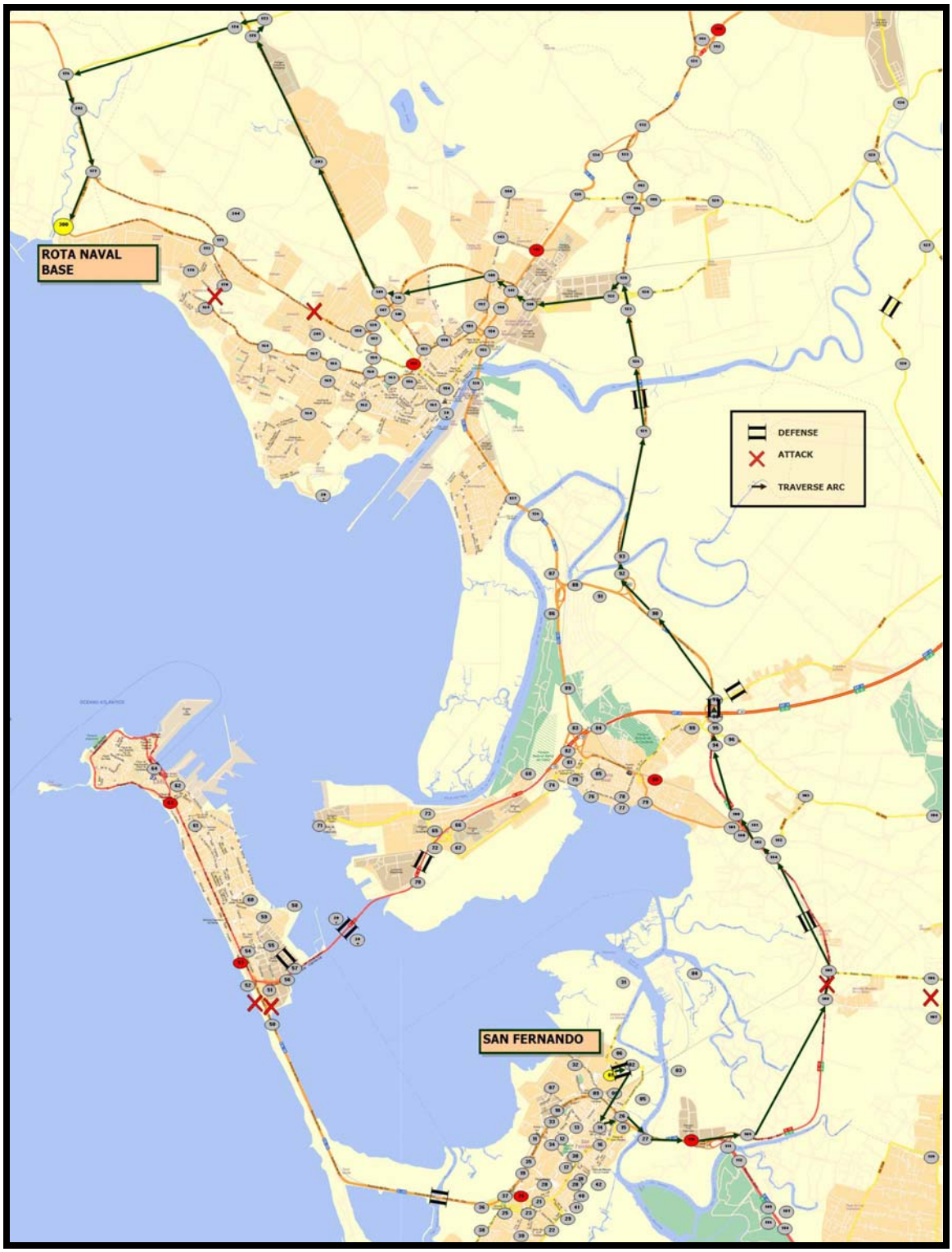

Figure 12. Area map showing the optimal solution to the tri-level problem given by *DAD* (Map from Michelin 2007).

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

Ahuja, R. K., Magnanti, T. L., and Orlin, J. B. (1993). *Network flows: Theory, algorithms, and applications*. Englewood Cliffs, N.J: Prentice Hall.

Birge, J. R., and Louveaux, F. (1997). *Introduction to stochastic programming*. New York: Springer.

Brown, G., Carlyle, M., Salmerón, J., and Wood, K. (2006). Defending critical infrastructure. *Interfaces*, **36**, pp. 530-547.

Brown, G., Carlyle, M., Salmerón, J., and Wood, R. K. (2005). Analyzing the vulnerability of critical infrastructure to attack, and planning defenses. *Tutorials in Operations Research. INFORMS 2005*, Institute for Operations Research and Management Science, Hanover, MD, pp. 102-123.

Consejería de Obras Públicas y Transportes - Junta de Andalucía. (2006). *Andalucia road network (red de carreteras de andalucia).* Retrieved February 17, 2007, from http://www.juntadeandalucia.es/obraspublicasytransportes/www/estaticas/carreteras/red_autonomica_carreteras/.

Cormican, K. J., Morton, D. P., and Wood, R. K. (1998). Stochastic network interdiction. *Operations Research*, **46**, p. 184.

Cournot, A. A. (1960). *Researches into the mathematical principles of the theory of wealth*, *1838* (Nathaniel T. Bacon Trans.). N.Y.: A. M. Kelley.

Dash Optimization Company. (2007). *Xpress-MP - leading optimization software.* Retrieved March 01, 2007, from http://www.dashoptimization.com/.

Department of Energy. (2007). Fossil energy: U.S. petroleum reserves. Retrieved July 29, 2007, from http://www.fe.doe.gov/programs/reserves/index.html.

Department of Homeland Security. (2002). *The national strategy for homeland security.* Retrieved February 16, 2007, from http://www.whitehouse.gov/homeland/book/.

Department of Homeland Security. (2006). *National infrastructure protection plan.* Retrieved February 16, 2007, from http://www.dhs.gov/xprevprot/programs/editorial_0827.shtm.

FEMA (Federal Emergency Management Agency). (2007). *FEMA 452 - risk assessment: A how-to guide to mitigate potential terrorist attacks.* Retrieved March 29, 2007, from http://www.fema.gov/plan/prevent/rms/rmsp452.shtm.

GAMS Development Corporation. (2007). *GAMS* (http://www.gams.com/ ed.) http://www.gams.com.

Google. (2007). *Google earth*. http://www.earth.google.com. Retrieved February 15, 2007.

Infanger G., and Morton D. (1996). Cut Sharing for Multistage Stochastic Linear Programs with Interstage Dependency. *Mathematical Programming*, **75**, pp. 241-256.

Israeli, E., and Wood, R. K. (2002). Shortest-path network interdiction. *Networks*, **40**, pp. 97-111.

Michelin. (2007). *ViaMichelin: Street map, maps, route finder, route planner, directions, road map, route map: Cadiz province road map.* Retrieved February 17, 2007, from http://www.viamichelin.com/viamichelin/int/tpl/hme/MaHomePage.htm.

Ministerio de Defensa (Spanish Departament of Defense). (2006). *Armada española. infantería de marina, tercio de armada, brigada de infantería de marina.* Retrieved February 26, 2007, from http://www.armada.mde.es/esp/BuquesUnidades/InfanteriaMarina/Tear/Brimar/uoe.asp.

Office Inspector General. Department of Homeland Security. (2006). *OIG 06-40. progress in developing the national asset database.* Retrieved February 20, 2007, from http://www.dhs.gov/xoig/assets/mgmtrpts/OIG_06-40_Jun06.pdf.

O'Neill, R. P. (1976). Nested decomposition of multistage convex programs. *SIAM Journal on Control and Optimization*, **14**, pp. 409-418.

Salmeron, J., Wood, R. K., and Baldick, R. (2004). *Optimizing electric grid design under asymmetric threat (II)*. Monterey, CA; Springfield, VA: Naval Postgraduate School; http://bosun.nps.edu/uhtbin/hyperion.exe/NPS-OR-04-001.pdf (1.07 MB).

Senate and House of Representatives of the United States of America. (2001). *USA PATRIOT act (H.R. 3162) uniting and strengthening america by providing appropriate tools required to intercept and obstruct terrorism.* Retrieved January 15, 2007, from http://www.epic.org/privacy/terrorism/hr3162.html.

Sidell, F. R. (2002). *Jane's chem-bio handbook* (2nd ed.). Alexandria, Va: Jane's Information Group.

Sun Microsystems, I. (2004). *Java™ 2 platform standard ed. 5.0*. http://java.sun.com/j2se/1.5.0/.

von Stackelberg, H. (1952). *The theory of the market economy* (T. A. Peacock Trans.). New York: Oxford University Press.

Wikipedia contributors. (2007). *San Fernando (Cádiz).* Retrieved February 26, 2007, from http://es.wikipedia.org/wiki/san_fernando_(cã*f*â¡diz)?oldid=7188499.

Wood, R. K. (1993). Deterministic network interdiction. *Mathematical and Computer Modelling*, 17, pp. 1-18.

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. Professor R. Kevin Wood
   Department of Operations Research
   Monterey, California

4. Professor Javier Salmeron
   Department of Operations Research
   Monterey, California